NOME CORSO

Fintech Software Developer

Unità Formativa (UF): Basi di dati SQL

Docente: Durando Giulio

Titolo argomento: Constraints di SQL SERVER

Constraints di SQL Server

I c**onstraints** (vincoli) sono regole che si applicano su una colonna o una tabella per evitare di inserire dati indesiderati nella tabella. È possibile specificare il limite del tipo di dati che può essere archiviato utilizzando i vincoli in una determinata colonna di una tabella. I vincoli possono essere a livello di colonna o di tabella. I vincoli a livello di colonna si applicano a una colonna e i vincoli a livello di tabella si applicano all'intera tabella.

È possibile definire i vincoli al momento della creazione della tabella utilizzando l'istruzione **CREATE TABLE**. Dopo aver creato una tabella, possiamo anche specificare i vincoli usando l'istruzione **ALTER TABLE**.

I vincoli comunemente usati in SQL sono:

- **NOT NULL**: garantisce che una colonna non possa avere un valore NULL
- **UNIQUE**: assicura che tutti i valori in una colonna siano diversi
- **PRIMARY KEY**: è la combinazione di NOT NULL e UNIQUE. Identifica in modo univoco ogni riga in una tabella
- **FOREIGN KEY**: impedisce le azioni che distruggerebbero i collegamenti tra le tabelle
- **CHECK**: garantisce che i valori in una colonna soddisfino una condizione specifica
- **DEFAULT**: imposta un valore predefinito per una colonna se non viene specificato alcun valore

Sintassi

CONSTRAINT
Specifica l'inizio della definizione di un vincolo.

constraint_name
Nome del vincolo. I nomi di vincolo devono essere conformi alle regole per gli identificatori, con l'eccezione che il nome non può iniziare con il simbolo di cancelletto (#). Se *constraint_name* viene omesso, al vincolo viene assegnato un nome generato dal sistema.

Nel caso di FOREIGN KEY devono essere specificati anche

schema_name
Nome dello schema a cui appartiene la tabella a cui il vincolo FOREIGN KEY fa riferimento.

referenced_table_name
Tabella a cui fa riferimento il vincolo FOREIGN KEY.

ref_column
Colonna tra parentesi a cui il nuovo vincolo FOREIGN KEY fa riferimento.

Nella creazione di una tabella per impostare i constraints si deve usare questa sintassi

```
Create TABLE Nome_Tabella
(
Column1 data_type(Size) Constraints_name,
Column2 data_type(Size) Constraints_name,
Column3 data_type(Size) Constraints_name,
);
```

dove

- **Nome_tabella:** è il nome della tabella che deve essere creata.
- **Data_type:** è il tipo di dati che può essere memorizzato nel campo.
- **Constraints_Name:** indica quale vincolo assegnare alla colonna (NOT NULL, UNIQUE, PRIMARY, ecc.)

Vincoli in SQL

NULL / NOT NULL

Per impostazione predefinita, una colonna può contenere valori NULL. Il vincolo NOT NULL impone a una colonna di NON accettare valori NULL. Ciò impone a un campo di contenere sempre un valore, il che significa che non è possibile inserire un nuovo record o aggiornare un record senza aggiungere un valore a questo campo. È possibile aggiungere colonne che non ammettono valori Null solo se a esse è associato un valore predefinito. Le nuove colonne che ammettono valori Null, ma a cui non è associato alcun valore predefinito contengono NULL per ogni riga della tabella.

Il seguente codice garantisce che le colonne "ID", "Cognome" e "Nome" NON accettino valori NULL quando viene creata la tabella "Persone":

```
CREATE TABLE Persone (
    ID int NOT NULL,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255) NOT NULL,
    Eta int
);
```

Per creare un vincolo NOT NULL sulla colonna "Età" quando la tabella "Persone" è già stata creata, si può utilizzare il seguente codice SQL:

```
ALTER TABLE Persons
MODIFY Eta int NOT NULL;
```

UNIQUE

Il vincolo UNIQUE garantisce che tutti i valori in una colonna siano diversi e che nella colonna non ci potranno mai essere due valori uguali. In una tabella, è possibile avere più di una colonna UNIQUE.

Il seguente codice crea un vincolo UNIQUE sulla colonna "ID" quando viene creata la tabella "Persone":

```
CREATE TABLE Persone (
    ID int NOT NULL UNIQUE,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int
);
```

Per consentire la creazione e l'assegnazione di un nome di un vincolo UNIQUE e per definire un vincolo UNIQUE su più colonne, utilizzare la seguente sintassi SQL:

```
CREATE TABLE Persone (
    ID int NOT NULL,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int,
    CONSTRAINT UC_Persone UNIQUE (ID,Cognome)
);
```

Per creare un vincolo UNIQUE sulla colonna "ID" quando la tabella è già stata creata, utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
ADD UNIQUE (ID);
```

Per consentire la creazione e l'assegnazione di un nome di un vincolo UNIQUE e per definire un vincolo UNIQUE su più colonne, utilizzare la seguente sintassi SQL:

```
ALTER TABLE Persone
ADD CONSTRAINT UC_Persona UNIQUE (ID,Cognome);
```

Per eliminare un vincolo UNIQUE, utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
DROP CONSTRAINT UC_Persona;
```

PRIMARY KEY

La chiave primaria viene utilizzata per identificare univocamente le righe dalla tabella. Se la tabella contiene la chiave primaria come campo, quel campo non può contenere valori **null** e poiché la chiave primaria identifica ciascuna riga in modo univoco, tutte le righe devono contenere valori univoci. In altre parole è una combinazione dei vincoli NOT NULL e UNIQUE. In una tabella può esserci solo una chiave primaria. Una chiave primaria può essere composta anche da più colonne. In questo caso le singole colonne possono avere valori non unique al proprio interno. L'importante è che la combinazione dei valori contenuti nelle singole colonne che fanno parte della chiave primaria abbia un valore unico

Il seguente codice SQL crea una CHIAVE PRIMARIA nella colonna "ID" quando viene creata la tabella "Persone":

```
CREATE TABLE Persone (
    ID int NOT NULL PRIMARY KEY,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int
);
```

Per consentire la creazione e l'assegnazione di un nome di un vincolo PRIMARY KEY e per definire un vincolo PRIMARY KEY su più colonne, utilizzare la seguente sintassi SQL:

```
CREATE TABLE Persone (
    ID int NOT NULL,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int,
    CONSTRAINT PK_Persona PRIMARY KEY (ID,Cognome)
);
```

Per creare un vincolo PRIMARY KEY sulla colonna "ID" quando la tabella è già stata creata, utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
ADD PRIMARY KEY (ID);
```

Per consentire la creazione e l'assegnazione di un nome ad un vincolo PRIMARY KEY e per definire un vincolo PRIMARY KEY su più colonne, utilizzare la seguente sintassi SQL:

```
ALTER TABLE Persone
ADD CONSTRAINT PK_Persone PRIMARY KEY (ID,Cognome);
```

N.B. Se si utilizza ALTER TABLE per aggiungere una chiave primaria, le colonne della chiave primaria devono essere state dichiarate NOT NULL (quando la tabella è stata creata per la prima volta).

Per eliminare un vincolo PRIMARY KEY, utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
DROP CONSTRAINT PK_Persone;
```

FOREIGN KEY

Per la creazione di una nuova tabella con una chiave esterna è richiesta l'autorizzazione **CREATE TABLE** per il database e l'autorizzazione **ALTER** per lo schema in cui viene creata la tabella. Per la creazione di una chiave esterna in una tabella esistente è richiesta l'autorizzazione **ALTER** per la tabella. Una foreign key (FK) è utilizzata per stabilire e applicare una relazione tra le due tabelle.

Limiti

- Un vincolo di chiave esterna non deve necessariamente essere collegato solo a un vincolo di chiave primaria in un'altra tabella. È anche possibile definire le chiavi esterne per fare riferimento alle colonne di un vincolo UNIQUE in un'altra tabella.
- I valori diversi da NULL immessi nella colonna di un vincolo FOREIGN KEY devono essere presenti nella colonna a cui viene fatto riferimento. In caso contrario, viene restituito un messaggio di errore di violazione della chiave esterna. Per assicurarsi che tutti i valori di un vincolo di chiave esterna composto vengano verificati, specificare NOT NULL in tutte le colonne coinvolte.
- I vincoli FOREIGN KEY possono fare riferimento a un'altra colonna nella stessa tabella. Questo tipo di vincolo viene definito autoreferenziale.
- Un vincolo FOREIGN KEY ***specificato a livello di colonna*** può includere una sola colonna di riferimento. Il tipo di dati di tale colonna deve essere uguale al tipo di dati della colonna in cui viene definito il vincolo.
- Un vincolo FOREIGN KEY ***specificato a livello di tabella*** (cioè su più colonne) deve includere lo stesso numero di colonne di riferimento di quelle presenti nell'elenco di colonne del vincolo. Il tipo di dati di ogni colonna di riferimento deve inoltre essere uguale a quello della colonna corrispondente nell'elenco di colonne.
- Una colonna di tipo **varchar(max)** può far parte di un vincolo FOREIGN KEY solo se anche la chiave primaria a cui fa riferimento è definita come tipo **varchar(max)** .

Il seguente codice SQL crea una foreign key nella colonna "IdPersona" quando viene creata la tabella "Ordini" che fa riferimento alla colonna IdPersona della tabella Persone:

```
CREATE TABLE Ordini (
   IDOrdine int NOT NULL PRIMARY KEY,
   NumeroOrdine int NOT NULL,
  IDPersona int FOREIGN KEY REFERENCES Persone(IDPersona)
);
```

Il seguente codice SQL crea una foreign key su più colonne colonna "IdPersona" quando viene creata la tabella "Ordini" che fa riferimento alla colonna IdPersona della tabella Persone:

```
CREATE TABLE Ordini (
   OrderID int NOT NULL,
   OrderNumber int NOT NULL,
   PersonID int,
   PRIMARY KEY (OrderID),
   CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
   REFERENCES Persons(PersonID)
);
```

Per creare un vincolo FOREIGN KEY sulla colonna "IDPersona" quando la tabella "Ordini" è già stata creata, utilizzare il seguente codice SQL:

```
ALTER TABLE Ordini
ADD FOREIGN KEY (IDPersona) REFERENCES Persone(IDPersona);
```

Per consentire la creazione e l'assegnazione di un nome ad un vincolo FOREIGN KEY e per definire un vincolo FOREIGN KEY su più colonne, utilizzare la seguente sintassi SQL:

```
ALTER TABLE Ordini
ADD CONSTRAINT FK_PersonOrder
FOREIGN KEY (IDPersona) REFERENCES Persone(IDPersona);
```

Per eliminare un vincolo FOREIGN KEY, utilizzare il seguente codice SQL:

```
ALTER TABLE Ordini
DROP CONSTRAINT FK_PersonOrder;
```

CHECK

Il vincolo CHECK viene utilizzato per limitare l'intervallo di valori che può essere inserito in una colonna. Se si definisce un vincolo CHECK su una colonna, verranno consentiti solo determinati valori per questa colonna. Se si definisce un vincolo CHECK su una tabella, è possibile limitare i valori in alcune colonne in base ai valori in altre colonne nella riga..

La sintassi per impostare un vincolo check quando viene creata una tabella è la seguente

```
CREATE TABLE Persone (
    ID int NOT NULL,
   Cognome varchar(255) NOT NULL,
   Nome varchar(255),
    Eta int CHECK (Eta>=18)
)
```

Per consentire la creazione di un vincolo CHECK e per definire un vincolo CHECK su più colonne, utilizzare la seguente sintassi SQL:

```
CREATE TABLE Persone (
    ID int NOT NULL,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int,
    Citta varchar(255),
    CONSTRAINT CHK_Person CHECK (Eta>=18 AND Citta='Roma')
);
```

Per creare un vincolo CHECK sulla colonna "Età" quando la tabella è già stata creata, utilizzare il seguente SQL

```
ALTER TABLE Persone
ADD CHECK (Eta>=18);
```

Per creare un vincolo CHECK e per definire un vincolo CHECK su più colonne, utilizzare la seguente sintassi SQL:

```
ALTER TABLE Persone
ADD CONSTRAINT CHK EtaPersona CHECK (Eta>=18 AND Citta='Roma');
```

per eliminare un vincolo CHECK, utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
DROP CONSTRAINT CHK_EtaPersona;
```

DEFAULT

Il vincolo DEFAULT viene utilizzato per impostare un valore predefinito per una colonna. Il valore predefinito verrà aggiunto a tutti i nuovi record, se non viene specificato alcun altro valore.

L'esempio SQL imposta un valore DEFAULT per la colonna "Città" quando viene creata la tabella "Persone":

```
CREATE TABLE Persone (
    ID int NOT NULL,
    Cognome varchar(255) NOT NULL,
    Nome varchar(255),
    Eta int,
    Citta varchar(255) DEFAULT 'Roma'
);
```

Il vincolo DEFAULT può essere utilizzato anche per inserire valori di sistema, utilizzando funzioni come GETDATE():

```
CREATE TABLE Ordini (
    ID int NOT NULL,
    NumeroOrdine int NOT NULL,
    Dataordin date DEFAULT GETDATE()
);
```

Per creare un vincolo DEFAULT sulla colonna "Città" quando la tabella è già stata creata, si deve utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
ADD CONSTRAINT df_Citta
DEFAULT 'Roma' FOR Citta;
```

Per eliminare un vincolo DEFAULT, si può utilizzare il seguente codice SQL:

```
ALTER TABLE Persone
ALTER COLUMN Citta DROP DEFAULT;
```