




Linguaggio SQL: fondamentali

Gestione delle tabelle


DBG



Gestione delle tabelle

- Creazione di una tabella
- Modifica della struttura di una tabella
- Cancellazione di una tabella
- Dizionario dei dati
- Integrità dei dati


DBG



Gestione delle tabelle

Creazione di una tabella

DBG



Creazione di una tabella (1/3)

➤ Si utilizza l'istruzione di SQL DDL (Data Definition Language)

CREATE TABLE

➤ Permette di

- definire tutti gli attributi (le colonne) della tabella
- definire vincoli di integrità sui dati della tabella

DBG

Creazione di una tabella (2/3)

```
CREATE TABLE NomeTabella  
(NomeAttributo Dominio [ValoreDiDefault ]  
 [Vincoli]  
{ , NomeAttributo Dominio [ValoreDiDefault ]  
 [Vincoli ]}  
AltriVincoli  
);
```



Creazione di una tabella (3/3)

➤ *Dominio*

- definisce il tipo di dato dell'attributo
 - domini predefiniti del linguaggio SQL (domini elementari)
 - domini definiti dall'utente a partire dai domini predefiniti

➤ *Vincoli*

- permette di specificare vincoli di integrità sull'attributo

➤ *AltriVincoli*

- permette di specificare vincoli di integrità di tipo generale sulla tabella



Definizione di domini (1/2)

➤ *ValoreDiDefault*

- permette di specificare il valore di default dell'attributo

DEFAULT

< *GenericoValore* | USER | CURRENT_USER | SESSION_USER | SYSTEM_USER | NULL >



Definizione di domini (2/2)

➤ *GenericoValore*

- valore compatibile con il dominio

➤ *USER

- identificativo dell'utente

➤ NULL

- valore di default di base



Domini elementari (1/6)

- Carattere: singoli caratteri o stringhe, anche di lunghezza variabile

CHARACTER [VARYING] [(Lunghezza)]
[CHARACTER SET NomeFamigliaCaratteri]

- abbreviato con VARCHAR

- Bit singoli (booleani) o stringhe di bit

BIT [VARYING] [(Lunghezza)]



Domini elementari (2/6)

- Numerici esatti

NUMERIC [(Precisione, Scala)]

DECIMAL [(Precisione, Scala)]

INTEGER

SMALLINT

- NUMERIC e DECIMAL sono numeri in base decimale



Domini elementari (3/6)

NUMERIC [(*Precisione*, *Scala*)]

DECIMAL [(*Precisione*, *Scala*)]

➤ Precisione

- numero totale di cifre (digits)
- per il dominio NUMERIC la precisione rappresenta un valore esatto
- per il dominio DECIMAL la precisione costituisce un requisito minimo



Domini elementari (3/6)

NUMERIC [(*Precisione*, *Scala*)]

DECIMAL [(*Precisione*, *Scala*)]

➤ Scala

- numero di cifre dopo la virgola

➤ Esempio: per il numero 123.45

- la precisione è 5, mentre la scala è 2



Domini elementari (4/6)

➤ Numerici approssimati

FLOAT [(*n*)]

REAL

DOUBLE PRECISION

➤ *n* specifica la precisione

- è il numero di bit utilizzati per memorizzare la mantissa di un numero float rappresentato in notazione scientifica
- è un valore compreso tra 1 e 53
- il valore di default è 53



Domini elementari (5/6)

INTERVAL *PrimaUnitàDiTempo*
[TO *UltimaUnitàDiTempo*]

➤ Le unità di tempo sono divise in due gruppi

- anno, mese
- giorno, ora, minuti, secondi

➤ Esempio: INTERVAL year TO month

- memorizza un periodo di tempo utilizzando i campi anno e mese

➤ Esempio: INTERVAL day TO second

- memorizza un periodo di tempo utilizzando i campi giorno, ore, minuti e secondi



Domini elementari (6/6)

- **TIMESTAMP [(Precisione)] [WITH TIME ZONE]**
 - memorizza i valori che specificano l'anno, il mese, il giorno, l'ora, i minuti, i secondi ed eventualmente la frazione di secondo
 - utilizza 19 caratteri più i caratteri per rappresentare la precisione
 - notazione
 - YYYY-MM-DD hh:mm:ss:p



Definizione di domini (1/2)

- Istruzione **CREATE DOMAIN**
 - definisce un dominio utilizzabile nelle definizioni di attributi

- Sintassi
`CREATE DOMAIN NomeDominio AS TipoDiDato
[ValoreDiDefault] [Vincolo]`

- *TipoDiDato* è un dominio elementare



Definizione di domini (2/2)

➤ Esempio

```
CREATE DOMAIN Voto AS SMALLINT
      DEFAULT NULL
      CHECK (Voto >= 18 and Voto <=30)
```



Definizione del DB fornitori prodotti

➤ Creazione della tabella fornitori

F

<u>CodF</u>	NomeF	NSoci	Sede
-------------	-------	-------	------

```
CREATE TABLE F (CodF CHAR(5),
                 NomeF CHAR(20),
                 NSoci SMALLINT,
                 Sede CHAR(15));
```

➤ Manca la definizione dei vincoli di integrità



Definizione del DB fornitori prodotti

➤ Creazione della tabella prodotti

P

<u>CodP</u>	NomeP	Colore	Taglia	Magazzino
-------------	-------	--------	--------	-----------

```
CREATE TABLE P (CodP CHAR(6),
                 NomeP CHAR(20),
                 Colore CHAR(6),
                 Taglia SMALLINT,
                 Magazzino CHAR(15));
```

➤ Manca la definizione dei vincoli di integrità



Definizione del DB fornitori prodotti

➤ Creazione della tabella forniture


FP

<u>CodF</u>	<u>CodP</u>	Qta
-------------	-------------	-----

```
CREATE TABLE FP (CodF CHAR(5),
                  CodP CHAR(6),
                  Qta INTEGER);
```

➤ Manca la definizione dei vincoli di integrità






Gestione delle tabelle

Modifica della struttura di una tabella

DBG



Istruzione ALTER TABLE (1/3)

⇒ Sono possibili le seguenti "alterazioni"

- aggiunta di una nuova colonna
- definizione di nuovo valore di default per una colonna (attributo) esistente
 - per esempio, sostituzione del precedente valore di default
- eliminazione di una colonna (attributo) esistente
- definizione di un nuovo vincolo di integrità
- eliminazione di un vincolo di integrità esistente

DBG

Istruzione ALTER TABLE (2/3)

ALTER TABLE *NomeTabella*

< ADD COLUMN <Definizione-Attributo> |

ALTER COLUMN *NomeAttributo*

< SET <Definizione-Valore-Default> | DROP DEFAULT > |

DROP COLUMN *NomeAttributo*

< CASCADE | RESTRICT > |

ADD CONSTRAINT [*NomeVincolo*]

< definizione-vincolo-unique > |

< definizione-vincolo-integrità-referenziale > |

< definizione-vincolo-check > |

DROP CONSTRAINT [*NomeVincolo*]

< CASCADE | RESTRICT >



Istruzione ALTER TABLE (3/3)

➤ RESTRICT

- l'elemento (colonna o vincolo) non è rimosso se è presente in qualche definizione di un altro elemento
- opzione di default

➤ CASCADE

- tutti gli elementi che dipendono da un elemento rimosso vengono rimossi, fino a quando non esistono più dipendenze non risolte (cioè non vi sono elementi nella cui definizione compaiono elementi che sono stati rimossi)



Istruzione ALTER TABLE: esempio n.1

- Aggiungere la colonna numero dipendenti alla tabella dei fornitori

F

CodF	NomeF	NSoci	Sede	NDipendenti
------	-------	-------	------	-------------

```
ALTER TABLE F  
ADD COLUMN NDipendenti SMALLINT;
```



Istruzione ALTER TABLE: esempio n.2

- Eliminare la colonna NSoci dalla tabella dei fornitori

F

CodF	NomeF	NSoci	Sede
------	-------	------------------	------

```
ALTER TABLE F  
DROP COLUMN NSoci RESTRICT;
```



Istruzione ALTER TABLE: esempio n.3

- Aggiungere il valore di default 0 alla colonna quantità della tabella delle forniture

FP

CodF	CodP	Qta
------	------	-----

```
ALTER TABLE FP  
ALTER COLUMN Qta SET DEFAULT 0;
```

DBG
M

Gestione delle tabelle

Cancellazione di una tabella

DBG
M

Cancellazione di una tabella

```
DROP TABLE NomeTabella  
[RESTRICT | CASCADE];
```

- Tutte le righe della tabella sono eliminate insieme alla tabella
- RESTRICT
 - la tabella non è rimossa se è presente in qualche definizione di tabella, vincolo o vista
 - opzione di default
- CASCADE
 - se la tabella compare in qualche definizione di vista anche questa è rimossa



Cancellazione di una tabella: esempio


- Cancellare la tabella fornitori

F

<u>CodF</u>	NomeF	NSoci	Sede
-------------	-------	-------	------

```
DROP TABLE F;
```






Gestione delle tabelle

Dizionario dei dati

DBG



Dizionario dei dati (1/2)

- I metadati sono informazioni (dati) sui dati
 - possono essere memorizzati in tabelle della base di dati
- Il dizionario dei dati contiene i metadati di una base di dati relazionale
 - contiene informazioni sugli oggetti della base di dati
 - è gestito direttamente dal DBMS relazionale
 - può essere interrogato con istruzioni SQL

DBG

Dizionario dei dati (2/2)

- Contiene diverse informazioni
 - descrizione di tutte le strutture (tabelle, indici, viste) della base di dati
 - stored procedure SQL
 - privilegi degli utenti
 - statistiche
 - sulle tabelle della base di dati
 - sugli indici della base di dati
 - sulle viste della base di dati
 - sulla crescita della base di dati



Informazioni sulle tabelle

- Il dizionario dei dati contiene per ogni tabella della base di dati
 - nome della tabella e struttura fisica del file in cui è memorizzata
 - nome e tipo di dato per ogni attributo
 - nome di tutti gli indici creati sulla tabella
 - vincoli di integrità



Tabelle del dizionario dati

- Le informazioni del dizionario dati sono memorizzate in alcune tabelle
 - ogni DBMS utilizza nomi diversi per tabelle diverse
- È possibile interrogare il dizionario dati mediante istruzioni SQL



Dizionario dati in Oracle (1/2)

- In Oracle sono definite 3 collezioni di informazioni per il dizionario dati
 - USER_*: metadati relativi ai dati dell'utente corrente
 - ALL_*: metadati relativi ai dati di tutti gli utenti
 - DBA_*: metadati delle tabelle di sistema



Dizionario dati in Oracle (2/2)

- USER_* contiene diverse tabelle e viste, tra le quali:
- USER_TABLES contiene metadati relativi alle tabelle dell'utente
 - USER_TAB_STATISTICS contiene le statistiche calcolate sulle tabelle dell'utente
 - USER_TAB_COL_STATISTICS contiene le statistiche calcolate sulle colonne delle tabelle dell'utente



Interrogazione del dizionario dati n.1

- Visualizzare il nome delle tabelle definite dall'utente e il numero di tuple memorizzate in ciascuna di esse

```
SELECT Table_Name, Num_Rows
FROM USER_TABLES;
```

R

Table_Name	Num_Rows
F	5
P	6
FP	12



Interrogazione del dizionario dati n.2 (1/2)

- Per ogni attributo della tabella delle forniture, visualizzare il nome dell'attributo, il numero di valori diversi e il numero di tuple che assumono valore NULL

```
SELECT Column_Name, Num_Distinct, Num_Nulls
FROM USER_TAB_COL_STATISTICS
WHERE Table_Name = 'FP'
ORDER BY Column_Name;
```



Interrogazione del dizionario dati n.2 (2/2)

```
SELECT Column_Name, Num_Distinct, Num_Nulls
FROM USER_TAB_COL_STATISTICS
WHERE Table_Name = 'FP'
ORDER BY Column_Name;
```

R

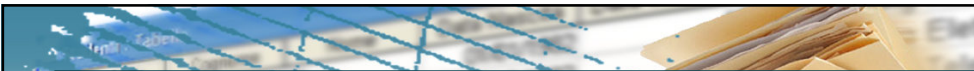
Column_Name	Num_Distinct	Num_Nulls
CodF	4	0
CodP	6	0
Qta	4	0






Gestione delle tabelle

Integrità dei dati



Vincoli di integrità

- ⊃ I dati all'interno di una base di dati sono corretti se soddisfano un insieme di regole di correttezza
 - le regole sono dette *vincoli di integrità*
 - esempio: $Q_{ta} \geq 0$
- ⊃ Le operazioni di modifica dei dati definiscono un nuovo stato della base dati, non necessariamente corretto



Verifica dell'integrità

- La verifica della correttezza dello stato di una base di dati può essere effettuata
 - dalle *procedure applicative*, che effettuano tutte le verifiche necessarie
 - mediante la definizione di *vincoli di integrità* sulle tabelle
 - mediante la definizione di *trigger*



Procedure applicative

- All'interno di ogni applicazione sono previste tutte le verifiche di correttezza necessarie
- Vantaggi
 - approccio molto efficiente
- Svantaggi
 - è possibile "aggirare" le verifiche interagendo direttamente con il DBMS
 - un errore di codifica può avere un effetto significativo sulla base di dati
 - la conoscenza delle regole di correttezza è tipicamente "nascosta" nelle applicazioni



Vincoli di integrità sulle tabelle (1/2)

- I vincoli di integrità sono
 - definiti nelle istruzioni CREATE o ALTER TABLE
 - memorizzati nel dizionario dati di sistema
- Durante l'esecuzione di qualunque operazione di modifica dei dati il DBMS verifica automaticamente che i vincoli siano osservati



Vincoli di integrità sulle tabelle (2/2)

- Vantaggi
 - definizione *dichiarativa* dei vincoli, la cui verifica è affidata al sistema
 - il dizionario dei dati descrive tutti i vincoli presenti nel sistema
 - unico punto centralizzato di verifica
 - impossibilità di aggirare la verifica dei vincoli



Vincoli di integrità sulle tabelle (2/2)

➤ Vantaggi

- definizione *dichiarativa* dei vincoli, la cui verifica è affidata al sistema
 - il dizionario dei dati descrive tutti i vincoli presenti nel sistema
- unico punto centralizzato di verifica
 - impossibilità di aggirare la verifica dei vincoli

➤ Svantaggi

- possono rallentare l'esecuzione delle applicazioni
- non è possibile definire tipologie arbitrarie di vincoli
 - esempio: vincoli su dati aggregati



Trigger (1/2)

➤ I trigger sono procedure eseguite in modo automatico quando si verificano opportune modifiche dei dati

- definiti nell'istruzione CREATE TRIGGER
- memorizzati nel dizionario dati del sistema

➤ Quando si verifica un evento di modifica dei dati sotto il controllo del trigger, la procedura viene eseguita automaticamente



Trigger (2/2)

➤ Vantaggi

- permettono di definire vincoli d'integrità di tipo complesso
 - normalmente usati insieme alla definizione di vincoli sulle tabelle
- unico punto centralizzato di verifica
 - impossibilità di aggirare la verifica dei vincoli

➤ Svantaggi

- applicativamente complessi
- possono rallentare l'esecuzione delle applicazioni



Riparazione delle violazioni

➤ Se un'applicazione tenta di eseguire un'operazione che violerebbe un vincolo, il sistema può

- impedire l'operazione, causando un errore di esecuzione dell'applicazione
- eseguire un'azione compensativa tale da raggiungere un nuovo stato corretto
 - esempio: quando si cancella un fornitore, cancellare anche tutte le sue forniture



Vincoli d'integrità in SQL-92

- Nello standard SQL-92 è stata introdotta la possibilità di specificare i vincoli di integrità in modo dichiarativo, affidando al sistema la verifica della loro consistenza
 - **vincoli di tabella**
 - restrizioni sui dati permessi nelle colonne di una tabella
 - **vincoli d'integrità referenziale**
 - gestione dei riferimenti tra tabelle diverse
 - basati sul concetto di chiave esterna



Vincoli di tabella (1/2)

- Sono definiti su una o più colonne di una tabella
- Sono definiti nelle istruzioni di creazione di
 - tabelle
 - domini
- Tipologie di vincolo
 - chiave primaria
 - ammissibilità del valore nullo
 - unicità
 - vincoli generali di tupla



Vincoli di tabella (2/2)

- Sono verificati dopo ogni istruzione SQL che opera sulla tabella soggetta al vincolo
 - inserimento di nuovi dati
 - modifica del valore di colonne soggette al vincolo
- Se il vincolo è violato, l'istruzione SQL che ha causato la violazione genera un errore di esecuzione



Chiave primaria

- La chiave primaria è un insieme di attributi che identifica in modo univoco le righe di una tabella
- Può essere specificata una sola chiave primaria per una tabella
- Definizione della chiave primaria
 - composta da un solo attributo

NomeAttributo Dominio PRIMARY KEY



Chiave primaria: esempio n. 1

```
CREATE TABLE F (CodF CHAR(5) PRIMARY KEY,  
NomeF CHAR(20),  
NSoci SMALLINT,  
Sede CHAR(15));
```



Chiave primaria

- La chiave primaria è un insieme di attributi che identifica in modo univoco le righe di una tabella
- Può essere specificata una sola chiave primaria per una tabella
- Definizione della chiave primaria
 - composta da uno o più attributi

PRIMARY KEY (*ElencoAttributi*)



Chiave primaria: esempio n. 2

```
CREATE TABLE FP (CodF CHAR(5),  
                  CodP CHAR(6),  
                  Qta  INTEGER  
                  PRIMARY KEY (CodF, CodP));
```



Ammissibilità del valore nullo

- Il valore NULL indica l'assenza di informazioni
- Quando è obbligatorio specificare sempre un valore per l'attributo

NomeAttributo Dominio NOT NULL

- il valore nullo non è ammesso



NOT NULL: esempio

```
CREATE TABLE F (CodF CHAR(5),  
NomeF CHAR(20) NOT NULL,  
NSoci SMALLINT,  
Sede CHAR(15));
```



Unicità

- ⊃ Un attributo o un insieme di attributi non può assumere lo stesso valore in righe diverse della tabella
 - per un solo attributo
NomeAttributo Dominio UNIQUE
 - per uno o più attributo
UNIQUE (*ElencoAttributi*)
- ⊃ È ammessa la ripetizione del valore NULL (considerato sempre diverso)



Chiave candidata

- La chiave candidata è un insieme di attributi che potrebbe assumere il ruolo di chiave primaria
 - è univoca
 - può non ammettere il valore nullo
- La combinazione **UNIQUE NOT NULL** permette di definire una chiave candidata che non ammette valori nulli

NomeAttributo Dominio **UNIQUE NOT NULL**



Unicità: esempio

```
CREATE TABLE P ( CodP      CHAR(6),  
                 NomeP    CHAR(20) NOT NULL UNIQUE,  
                 Colore   CHAR(6),  
                 Taglia   SMALLINT,  
                 Magazzino CHAR(15));
```



Vincoli generali di tupla

- Permettono di esprimere condizioni di tipo generale su ogni tupla
 - vincoli di tupla o di dominio
NomeAttributo Dominio CHECK (Condizione)
 - possono essere indicati come condizione i predicati specificabili nella clausola WHERE
- La base di dati è corretta se la condizione è vera



Vincoli generali di tupla: esempio

```
CREATE TABLE F (CodF CHAR(5) PRIMARY KEY,  
NomeF CHAR(20) NOT NULL,  
NSoci SMALLINT  
CHECK (NSoci>0),  
Sede CHAR(15));
```



Vincoli d'integrità referenziale

➤ Permettono di gestire il legame tra tabelle mediante il valore di attributi

➤ Esempio

F			
<u>CodF</u>	NomeF	NSoci	Sede

FP		
<u>CodF</u>	<u>CodP</u>	Qta

- la colonna CodF di FP può assumere valori già presenti nella colonna CodF di F
 - CodF in FP: colonna referenziante (o chiave esterna)
 - CodF in F: colonna referenziata (tipicamente la chiave primaria)



Definizione della chiave esterna

➤ La chiave esterna è definita nell'istruzione CREATE TABLE della tabella referenziante

```
FOREIGN KEY (ElencoAttributiReferenzianti)
REFERENCES
NomeTabella [(ElencoAttributiReferenziati)]
```

➤ Se gli attributi referenziati hanno lo stesso nome di quelli referenzianti, non è obbligatorio specificarli



Definizione della chiave esterna: esempio

```
CREATE TABLE FP (CodF CHAR(5),  
                CodP CHAR(6),  
                Qta INTEGER,  
                PRIMARY KEY (CodF, CodP),  
                FOREIGN KEY (CodF)  
                    REFERENCES F(CodF),  
                FOREIGN KEY (CodP)  
                    REFERENCES P(CodP));
```



Gestione dei vincoli: esempio n.1

- Tabella FP (referenziante)
 - insert (nuova tupla) -> No
 - update (CodF) -> No
 - delete (tupla) -> Ok
- Tabella F (referenziata)
 - insert (nuova tupla) -> Ok
 - update (CodF) -> aggiornare in cascata (cascade)
 - delete (tupla) -> aggiornare in cascata (cascade)
impedire l'azione (no action)



Gestione dei vincoli: esempio n.2 (1/3)

- Impiegati (Matr, NomeI, Residenza, DNum)
- Dipartimenti (DNum, DNome, Sede)



Gestione dei vincoli: esempio n.2 (2/3)

- Impiegati (referenziante)



Gestione dei vincoli: esempio n.2 (2/3)

➤ Impiegati (referenziante)

- insert (nuova tupla) -> No
- update (DNum) -> No
- delete (tupla) -> Ok



Gestione dei vincoli: esempio n.2 (3/3)

➤ Dipartimenti (referenziata)

- insert (nuova tupla) -> Ok
- update (DNum) -> aggiornare in cascata (cascade)
- delete (tupla) -> aggiornare in cascata (cascade)
impedire l'azione (no action)
impostare a valore ignoto (set null)
impostare a valore di default (set default)



Politiche di gestione dei vincoli (1/3)

- I vincoli d'integrità sono verificati dopo ogni istruzione SQL che potrebbe causarne la violazione
- Non sono ammesse operazioni di inserimento e modifica della tabella referenziante che violino il vincolo



Politiche di gestione dei vincoli (2/3)

- Operazioni di modifica o cancellazione dalla tabella referenziata causano sulla tabella referenziante:
 - CASCADE: propagazione dell'operazione di aggiornamento o cancellazione
 - SET NULL/DEFAULT: null o valore di default in tutte le colonne delle tuple che hanno valori non più presenti nella tabella referenziata
 - NO ACTION: non si esegue l'azione invalidante



Politiche di gestione dei vincoli (3/3)

- Nell'istruzione CREATE TABLE della tabella referenziata

```
FOREIGN KEY (ElencoAttributiReferenzianti)
REFERENCES
NomeTabella [(ElencoAttributiReferenziati)]
[ON UPDATE
<CASCADE | SET DEFAULT | SET NULL |
NO ACTION>]
[ON DELETE
<CASCADE | SET DEFAULT | SET NULL |
NO ACTION>]
```



Base dati di esempio (1/4)

- DB forniture prodotti

- **tabella P: descrive i prodotti disponibili**
 - chiave primaria: CodP
 - nome prodotto non può assumere valori nulli o duplicati
 - la taglia è sempre maggiore di zero
- **tabella F: descrive i fornitori**
 - chiave primaria: CodF
 - nome fornitore non può assumere valori nulli o duplicati
 - numero dei soci è sempre maggiore di zero



Base dati di esempio (1/4)

➤ DB forniture prodotti

- tabella FP: descrive le forniture, mettendo in relazione i prodotti con i fornitori che li forniscono
 - chiave primaria: (CodF, CodP)
 - quantità non può assumere il valore null ed è maggiore di zero
 - vincoli di integrità referenziale



Base dati di esempio (2/4)

```
CREATE TABLE P ( CodP      CHAR(6) PRIMARY KEY,  
                 NomeP    CHAR(20) NOT NULL UNIQUE,  
                 Colore   CHAR(6),  
                 Taglia    SMALLINT  
                 CHECK (Taglia > 0),  
                 Magazzino CHAR(15));
```



Base dati di esempio (3/4)

```
CREATE TABLE F (CodF CHAR(5) PRIMARY KEY,  
NomeF CHAR(20) NOT NULL,  
NSoci SMALLINT  
CHECK (NSoci>0),  
Sede CHAR(15));
```



Base dati di esempio (4/4)

```
CREATE TABLE FP (CodF CHAR(5),  
CodP CHAR(6),  
Qta INTEGER  
CHECK (Qta IS NOT NULL and Qta>0),  
PRIMARY KEY (CodF, CodP),  
FOREIGN KEY (CodF)  
REFERENCES F(CodF)  
ON DELETE NO ACTION  
ON UPDATE CASCADE,  
FOREIGN KEY (CodP)  
REFERENCES P(CodP)  
ON DELETE NO ACTION  
ON UPDATE CASCADE);
```

