Politecnico di Torino

Gestione delle tabelle

Linguaggio SQL

Linguaggio SQL: gestione delle tabelle

- Creazione tabelle
- Modifica delle tabelle
- Cancellazione delle tabelle
- Integrità dei dati

Creazione delle tabelle

Gestione delle tabelle

CREATE

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

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

- *ValoreDiDefault*
  - permette di specificare il valore di default dell'attributo
- GenericoValore
  - valore compatibile con il dominio
- *USER
  - identificativo dell'utente
- NULL
  - valore di default di base

Domini elementari

Tipologia di dato	SQL
Testo	CHARACTER [VARYING] [(*Lunghezza*)] [CHARACTER SET *NomeFamigliaCaratteri*] VARCHAR (Lunghezza) TEXT
Binario	BIT [VARYING] [(*Lunghezza*)] BLOB BINARY
Booleano	BOOLEAN
Numeri interi	INTEGER SMALLINT BIGINT
Numeri reali	NUMERIC [(*Precisione, Scala*)] DECIMAL [(*Precisione, Scala*)] FLOAT [(*n*)] REAL DOUBLE PRECISION

Domini elementari

Tipologia di dato	SQL
Tempo	TIMESTAMP *[(Precisione)]* [WITH TIME ZONE] DATE DATETIME
JSON	JSON
Spaziali	SDO_GEOMETRY GEOMETRY POINT LINESTRING POLYGON

La definizione delle tipologie di dato in SQL differisce dal DBMS usato

Definizione database fornitori-prodotti

- Creazione della tabella fornitori

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

- Creazione della tabella forniture

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

- Creazione della tabella prodotti

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

Manca la definizione dei vincoli di integrità

Modifica delle tabelle

Gestione delle tabelle

ALTER TABLE

- 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

```
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 >
```

- RESTRICT (opzione di default)
  - l’elemento (colonna o vincolo) non è rimosso se è presente in qualche definizione di un altro elemento
- CASCADE
  - tutti gli elementi che dipendono da un elemento rimosso vengono rimossi, fino a quando non esistono più dipendenze non risolte

Esempi: modifica struttura della tabella

- Aggiungere la colonna numero dipendenti alla tabella dei fornitori

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

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

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

- Eliminare la colonna NSoci dalla tabella dei fornitori

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

Cancellazione delle tabelle

Gestione delle tabelle

DROP TABLE

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

Dizionario dei dati

Dizionario dei dati

- 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
- 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

- 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
- 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

- 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;
```

R

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

Integrità dei dati

Gestione delle tabelle

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: Qta >=0
- Le operazioni di modifica dei dati definiscono un nuovo stato della base dati, non necessariamente corretto
- 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 “flessibile ”

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

- 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

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

- 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

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

- Possibilità di specificare i vincoli di integrità in modo dichiarativo
- Si affida al sistema la verifica della loro consistenza
- Tipologie di vincoli:
  - 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

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

  - composta da uno o più attributi

    PRIMARY KEY (*ElencoAttributi*)

Esempi chiave primaria

un solo attributo

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

uno o più attributi

```
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

Esempio: NOT NULL

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

UNIQUE

- 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

Esempio: UNIQUE

```
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

Esempio: vincoli generali di tupla

```
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
- 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

Esempio: Definizione della chiave esterna

```
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));
```

Politiche di gestione dei vincoli

- 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
- Nell'istruzione CREATE TABLE della tabella referenziante

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

- 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

Esempio: 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
- 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

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)

Esempio: Insert, Delete, Update su tabella FP

F

CodF	FNome	Città
F1	Smith	London
F2	Jones	Paris
F3	Blake	Paris
F4	Clark	London
F5	Adams	Athens

INSERT

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F2	P1	300
F2	P2	400
F3	P2	200
F4	P5	400

Insert

F1	P1	300

OK

Insert

F10	P1	300

NO

UPDATE

FP

CodF	CodP	Qta	
F1	P1	300	
~~F1~~ F5	P2	200	OK
F1	P3	400	
F1	P4	200	
F2	P1	300	
~~F2~~ *F10*	P2	400	NO
F3	P2	200	
F4	P5	400	

DELETE

FP

CodF	CodP	Qta	
~~F1~~	~~P1~~	~~300~~	OK
F1	P2	200	
F1	P3	400	
F1	P4	200	
~~F2~~	~~P1~~	~~300~~	OK
F2	P2	400	
F3	P2	200	
~~F4~~	~~P5~~	~~400~~	OK

Esempio: Insert in F

F

CodF	FNome	Città
F1	Smith	London
F2	Jones	Paris
F3	Blake	Paris
F4	Clark	London
F5	Adams	Athens

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F2	P1	300
F2	P2	400
F3	P2	200
F4	P5	400

Insert

F10	Blake	Torino

OK

Esempio: Delete from F

F

CodF	FNome	Città
F1	Smith	London
F2	Jones	Paris
F3	Blake	Paris
~~F4~~	~~Clark~~	~~London~~
F5	Adams	Athens

CASCADE
operazione
delete è
propagata

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F2	P1	300
F2	P2	400
F3	P2	200
~~F4~~	~~P5~~	~~400~~

NO ACTION
operazione
delete non viene
eseguita

SET NULL/DEFAULT
non è applicabile in questo caso. Tale opzione imposta un valore nullo o predefinito nelle colonne per le tuple i cui valori non sono più presenti nella tabella di riferimento

Esempio: Update di F

F

CodF	FNome	Città
F1	Smith	London
F2	Jones	Paris
F3	Blake	Paris
~~F4~~ *F6*	Clark	London
F5	Adams	Athens

CASCADE operazione update è propagata

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F2	P1	300
F2	P2	400
F3	P2	200
~~F4~~ *F6*	P5	400

NO ACTION operazione delete non viene eseguita

L'opzione SET NULL/DEFAULT non è applicabile in questo caso. Tale opzione imposta un valore nullo o predefinito nelle colonne per le tuple i cui valori non sono più presenti nella tabella di riferimento

Esempio SQL: DB forniture prodotti

```
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));

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

```
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);
```

Gestione dei vincoli: esempio n.2

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

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