Linguaggio SQL: fondamenti

Gestione delle tabelle

A.a. 2019/20

Gestione delle tabelle

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

Gestione delle tabelle

Creazione di una tabella

A.a. 2019/20

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

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, CURRENT_USER, SESSION_USER, …
  - identificativo dell'utente
- NULL
  - valore di default di base

Domini elementari (1/6)

- Carattere: singoli caratteri o stringhe di lunghezza fissa o variabile entro il limite indicato

```
CHAR[(Lunghezza fissa)]
        [CHARACTER SET NomeFamigliaCaratteri]

VARCHAR[(Lunghezza massima)]
        [CHARACTER SET NomeFamigliaCaratteri]
```

- Bit singoli (booleani) o stringhe di bit

```
BIT [VARYING] [(Lunghezza)]
```

Domini elementari (2/6)

- Numerici esatti

  NUMERIC [(*Precisione, Scala*)]
  DECIMAL [(*Precisione, Scala*)]
  INTEGER
  SMALLINT

- *Precisione, Scala* in NUMERIC e DECIMAL sono valori riferiti alla rappresentazione in base 10

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

CodF	NomeF	NSoci	Sede

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

Definizione del DB fornitori prodotti

- Creazione della tabella prodotti

P

CodP	NomeP	Colore	Taglia	Magazzino

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

Definizione del DB fornitori prodotti

- Creazione della tabella forniture

FP

CodF	CodP	Qta

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

- Manca la definizione dei vincoli di integrità referenziale

Gestione delle tabelle

Modifica della struttura di una tabella

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

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


```
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;
```

Gestione delle tabelle

Cancellazione di una tabella

A.a. 2019/20

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

CodF	NomeF	NSoci	Sede

```
DROP TABLE F;
```

Gestione delle tabelle

Integrità dei dati

A.a. 2019/20

DBMG

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

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 e non viene eseguita

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 Tipo 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

CodF	NomeF	NSoci	Sede

FP

CodF	CodP	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*)]

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));
```

Politiche di gestione dei vincoli (1)

- I vincoli d'integrità sono verificati dopo ogni istruzione SQL che potrebbe causarne la violazione
- Vincoli di integrità referenziale:
  - *non sono ammesse* operazioni di inserimento e modifica della tabella referenziante che violino il vincolo
  - se il progettista ha previsto nel DB *azioni compensative* nel caso di potenziale violazione di un vincolo di integrità referenziale, queste azioni vengono eseguite senza comportare alcun errore di esecuzione (i vincoli sono comunque *mantenuti*)

Politiche di gestione dei vincoli (2)

- *Azioni compensative* nel caso di istruzioni che comportino la potenziale violazione di un vinvolo di integrità referenziale:
  - azioni possibili nel caso di *modifica di dati* nella tabella referenziata (UPDATE)
  - azioni possibili nel caso di *eliminazione di tuple* nella tabella referenziata (DELETE)

Politiche di gestione dei vincoli (3)

- Azioni compensative nel caso di MODIFICA *di dati* nella tabella referenziata (UPDATE)

CASCADE	propagazione *in cascata*, ai dati *referenzianti*, dell'operazione di aggiornamento
SET NULL	assegnazione del valore NULL a tutti gli attributi *referenzianti* con valori non più presenti nella tabella *referenziata*
SET DEFAULT	assegnazione del *valore di default* a tutti gli attributi *referenzianti* con valori non più presenti nella tabella referenziata
NO ACTION:	azione di modifica *non eseguita*

Nota: queste clausole possono variare a seconda del DBMS

Politiche di gestione dei vincoli (3)

- Azioni compensative nel caso di ELIMINAZIONE *di tuple* nella tabella referenziata (DELETE)

CASCADE	propagazione *in cascata*, alle tuple *referenzianti*, dell'operazione di eliminazione
SET NULL	assegnazione del valore NULL a tutti gli attributi *referenzianti* con valori non più presenti nella tabella *referenziata*
SET DEFAULT	assegnazione del *valore di default* a tutti gli attributi *referenzianti* con valori non più presenti nella tabella referenziata
NO ACTION:	azione di eliminazione *non eseguita*

Nota: queste clausole possono variare a seconda del DBMS

Politiche di gestione dei vincoli (4)

- 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>]
```

DB di esempio Forniture (1)

- 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

DB di esempio Forniture (2)

- 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

DB di esempio Forniture (3)

```
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));
```

DB di esempio Forniture (4)

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

DB di esempio Forniture (5)

```
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);
```