52
SQL: CONCETTI BASE

52.1
Introduzione

Sql, che è il linguaggio standard per la programmazione dei database (d’ora in poi, db) relazionali.

Userò soltanto le istruzioni basiche; voi (negli esercizi) potrete usare soltanto le istruzioni viste a lezione.

L’Sql (basico) non è case sensitive.

Io seguirò le seguenti convenzioni:

~ Ogni parola che indica una tabella oppure che indica una parola chiave sarà scritta in maiuscolo.

~ Le costanti booleane false e true saranno scritte in minuscolo.

~ Ogni parola che indica un attributo, sarà scritta con l’iniziale di ogni parola in maiuscolo e con ogni altra lettera in minuscolo. (Es: NStudePrese.)

52.2
Database di riferimento

Ogni esempio o esercizio di questo capitolo si riferisce al seguente database, tranne quando indicherò diversamente.

```
DIPARTIMENTI(NomeDipa, Citta, MatriImpieCassiere*)
    MatriImpieCassiere = NULL => Il dipartimento attualmente non ha nessun
      cassiere.
    IR: MatriImpieCassiere -> IMPIEGATI.MatriImpie

IMPIEGATI(MatriImpie, Cognome, Nome, Citta, Stipendio, NomeDipa*, Ufficio*,
  MatriImpieCapo*)
    NomeDipa = NULL => L’impiegato attualmente non è assegnato in nessun
        dipartimento.
    Ufficio = NULL => L’impiegato attualmente non è assegnato in nessun
        ufficio.
    MatriImpeCapo = NULL => L’impiegato attualmente non ha un capo.
    IR: NomeDipa -> DIPARTIMENTI.NomeDipa
    IR: MatriImpieCapo -> IMPIEGATI.MatriImpie
```

Ad es., il database può contenere questi valori:

DIPARTIMENTI

NomeDipa	Citta	MatriImpieCassiere
HW	MI	3
SW	PD	NULL
CE	PA	1

IMPIEGATI

MatriImpie	Cognome	Nome	Citta	Stipendio	NomeDipa	Ufficio	MatriImpieCapo
1	A	X	PD	50	SW	1	NULL
2	A	Y	VI	80	SW	2	NULL
3	B	X	MI	15	HW	2	1
4	C	Y	MI	60	NULL	NULL	1
5	C	Y	PD	20	SW	2	2
6	D	Z	PD	90	NULL	NULL	NULL
7	E	A	VI	90	SW	1	4

52.3 Definizione dei dati in Sql

52.3.1 Altri domini che useremo

Interi autoincrementanti

Es: Matricola, Identificatore in un registro.

NB L’operazione di addizione sugli autoincrementanti non è utile. (Invece, lo è sugli interi “normali”.)

La sintassi standard per definire per un intero autoincrementante è:

```
NomeAttributo INTEGER PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY
```

NB La sintassi suddetta, pur essendo standard, non è ancora molto “diffusa”.

52.3.2 Definizione dei domini

```
CREATE DOMAIN Voto AS SMALLINT DEFAULT 18 CHECK (VALUE >= 18 AND VALUE <= 30);
```

Se inserisco un valore esterno all’intervallo di un dominio, il DBMS rifiuta l’inserimento.

52.3.3 Definizione dei vincoli

La presenza dei vincoli è finalizzata a evitare inserimenti errati dell’utente.

Naturalmente, è impossibile che il DBMS conosca completamente il pensiero dell’utente e che quindi possa sempre rifiutare gli inserimenti errati dell’utente. Ad es. se un utente inserisce la città di residenza errata, il DBMS non può accorgersene.

Però, il DBMS può comprendere se un inserimento di dati fatto dall’utente è incoerente con i vincoli che il programmatore aveva precedentemente definito. In questo caso, il DBMS rifiuta l’inserimento.

Le categorie seguenti di vincoli seguenti (che sono i più importanti) sono definibili direttamente in Sql:

~ Di dominio
  Abbiamo visto precedentemente come definire un dominio. Dopodiché, possiamo definire un attributo vincolato a questo dominio mediante la sintassi:
  ```
  Attributo Dominio
  ```

~ Attributo obbligatorio
  Per default, un attributo è opzionale, tranne quando appartiene alla chiave primaria.
  Per vincolare un attributo a essere obbligatorio, dobbiamo usare `NOT NULL` mediante la sintassi:
  `Attributo Dominio NOT NULL`

~ Di chiave primaria

~ Di chiave candidata

~ Di integrità referenziale

Esistono altre categorie (più complesse) di vincoli; per definirli, dovremo usare le asserzioni e i trigger, che vedremo in un capitolo seguente.

Adesso vediamo la definizione dei vincoli di chiave primaria, chiave candidata, integrità referenziale.

Sintassi

Per definire i vincoli di chiave primaria, oppure di chiave candidata, oppure di integrità referenziale, la sintassi si differenzia a seconda dei due casi seguenti:

~ il vincolo è associato a un singolo attributo

~ il vincolo è associato a due o più attributi.

Sintetizzo le varie situazioni nello schema seguente.

Nel caso di IR, indico in questo modo i vincoli di IR che devo tradurre in Sql:

~ IR: `Attributo -> TABE_PRINCI.Attributo`, se il vincolo è associato a un singolo attributo;

~ IR: `{Attributo1, Attributo1} -> {TABE_PRINCI.Attributo}`, se il vincolo è associato a due attributi.

Tipo di vincolo	N. di attributi	Sintassi
Chiave (primaria)	1	Quando si definisce l’attributo, si aggiunge `Attributo Dominio PRIMARY KEY`
Chiave (primaria)	>= 2	Dopo avere definito ogni attributo si aggiunge `PRIMARY KEY (Attributo1, Attributo2)`
Chiave candidata (non primaria)	1	Quando si definisce l’attributo, si aggiunge `Attributo Dominio UNIQUE`
Chiave candidata (non primaria)	>= 2	Dopo avere definito ogni attributo si aggiunge `UNIQUE (Attributo1, Attributo2)`
Integrità referenziale	1	Quando si definisce l’attributo, si aggiunge `Attributo Dominio REFERENCES TABE_PRINCI(Attributo)`
Integrità referenziale	>= 2	Dopo avere definito ogni attributo si aggiunge `FOREIGN KEY (Attributo1, Attributo2) REFERENCES TABE_PRINCI(Attributo1, Attributo2)`