Il linguaggio SQL: le viste

Angelo Montanari

Dipartimento di Matematica e Informatica
Università di Udine

Introduzione

Livello *esterno* (o delle viste): mettere a disposizione degli utenti rappresentazioni diverse degli stessi dati.

Distinzione tra relazioni/tabelle di base e tabelle *derivate*.

Due *tipologie* di tabelle derivate:

- *relazioni virtuali* (o **viste**)
  - relazioni definite per mezzo di interrogazioni
  - dotate di schema, ma prive di istanza
  - utilizzabili nelle interrogazioni come le tabelle di base
- *viste materializzate*, tabelle derivate il cui contenuto (istanza) viene effettivamente memorizzato nella base di dati (problema: mantenimento dell'*allineamento* con le tabelle di base)

Un semplice esempio

Sia data una base di dati contenente le seguenti relazioni:

$R(A, B, C)$, $S(C, D, E)$ e $T(F, G)$.

Sia V la vista definita dall'interrogazione

$$V = \sigma_{A>D}(R \bowtie S)$$

L'interrogazione

$$V \bowtie_{B=F} T$$

viene eseguita sostituendo alla vista V la sua definizione:

$$(\sigma_{A>D}(R \bowtie S)) \bowtie_{B=F} T$$

Uso delle viste: vantaggi

- Selezionare le sole componenti di interesse

$$\Pi_{NOME,COGNOME,DNOME}(IMPIEGATO \bowtie_{DIP=DNUMERO} DIPARTIMENTO)$$

- Esprimere in modo compatto espressioni molto complesse, con sottoespressioni ripetute
- Formulare interrogazioni altrimenti non esprimibili
- Introdurre meccanismi di protezione della privatezza (autorizzazioni di accesso rispetto alle viste)
- In presenza di ristrutturazioni, inserire viste che corrispondono a relazioni non più presenti, ma ricavabili dalle nuove relazioni

Le viste in SQL - 1

Le viste vengono definite in SQL nel modo seguente.

Sintassi: CREATE VIEW NomeVista [(ListaAttributi)] AS
SELECT-SQL
[WITH [LOCAL | CASCADED] CHECK OPTION]

L'interrogazione deve restituire un insieme di attributi compatibile con gli attributi dichiarati nello schema della vista.

Esempio. Definire una vista IMPIEGATI5 che contiene tutti e soli gli impiegati del dipartimento 5 che guadagnano più di 10000 euro.

```
CREATE VIEW IMPIEGATI5(CF5, NOME5, COGNOME5, STIP5) AS
    SELECT CF, NOME, COGNOME, STIPENDIO
    FROM   IMPIEGATO
    WHERE DIP = 5 AND STIPENDIO > 10000
```

Le viste in SQL - 2

A partire dalla vista IMPIEGATI5, può essere costruita una seconda vista che contiene tutti e soli gli impiegati del dipartimento 5 con uno stipendio compreso tra 10000 e 25000 (impiegati poveri).

```
CREATE VIEW IMPIEGATI5POVERI AS
    SELECT *
    FROM   IMPIEGATI5
    WHERE STIPENDIO < 25000
    WITH CHECK OPTION
```

La dichiarazione WITH CHECK OPTION, con l'eventuale qualifica LOCAL o CASCADED (CASCADED è il default) concerne la possibilità di effettuare o meno operazioni di modifica di una vista.

Interrogazioni SQL e viste - 1

Le viste possono essere utilizzate per la stesura di interrogazioni altrimenti non esprimibili in SQL, aumentando in tal modo il potere espressivo del linguaggio.

Utilizzo di più **funzioni aggregate in cascata** (o uso non elementare dell'operatore di unione).

Esempio. Determinare il dipartimento/i caratterizzato dal massimo della somma degli stipendi pagati agli afferenti.

```
CREATE VIEW DIPSTIP(DIP, SOMMASTIPENDI) AS
    SELECT     DIP, SUM(STIPENDIO)
    FROM       IMPIEGATO
    GROUP BY DIP

SELECT DIP
FROM   DIPSTIP
WHERE SOMMASTIPENDI >= ALL ( SELECT   SOMMASTIPENDI
                             FROM     DIPSTIP)
```

Interrogazioni SQL e viste - 2

Soluzione alternativa.

```
SELECT     DIP
FROM       IMPIEGATO
GROUP BY DIP
HAVING     SUM(STIPENDIO) >= ALL (SELECT     SUM(STIPENDIO)
                                  FROM       IMPIEGATO
                                  GROUP BY DIP)
```

Questa seconda soluzione non è supportata da tutti i sistemi (alcuni sistemi impongono la restrizione che la condizione associata alla clausola HAVING non contenga interrogazioni nidificate).

Interrogazioni SQL e viste - 3

Esempio. Limitatamente agli impiegati con almeno una persona a carico, determinare il numero medio e massimo di persone a carico.

```
CREATE VIEW CARICOIMP(IMP, CARICO) AS
    SELECT    IMP, COUNT(*)
    FROM      PERSONA_A_CARICO
    GROUP BY IMP

SELECT    AVG(CARICO), MAX(CARICO)
FROM      CARICOIMP
```

Soluzione alternativa (scorretta)

```
SELECT    AVG(COUNT(*)), MAX(COUNT(*))
FROM      PERSONA_A_CARICO
GROUP BY IMP
```

In generale, la valutazione (in cascata) di due diversi operatori avviene a livelli diversi di aggregazione, ma è ammessa una sola occorrenza della clausola GROUP BY per ogni interrogazione.

Viste ricorsive in SQL-3

SQL-2 non ammette dipendenze ricorsive tra viste, né immediate né transitive.

SQL-3 mutua dal linguaggio Datalog la modalità di definizione di viste ricorsive.

Esempio. Trovare i supervisori diretti o indiretti di Giovanni Rossi.

```
WITH RECURSIVE HASUPERIORE(IMPIEGATO,SUPERIORE) AS
     (( SELECT CF, SUPERVISORE
        FROM   IMPIEGATO )
        UNION
     (  SELECT I1.CF, I2.SUPERIORE
        FROM   IMPIEGATO I1, HASUPERIORE I2
        WHERE I1.SUPERVISORE = I2.IMPIEGATO))

SELECT     CF, SUPERIORE
FROM       (IMPIEGATO JOIN HASUPERIORE on
           IMPIEGATO.CF = HASUPERIORE.IMPIEGATO)
WHERE      NOME = 'GIOVANNI' AND COGNOME = 'ROSSI'
```

Viste e operazioni di modifica

Su alcune viste si possono effettuare operazioni di modifica che vengono trasformate in operazioni corrispondenti sulle tabelle di base da cui tali viste dipendono.

Non sempre è possibile ricondurre in modo univoco le modifiche sulle viste a modifiche delle tabelle di base.

Esempio.

```
CREATE VIEW TABELLA_LAVORA_A_RIVISTA AS
    SELECT NOME, COGNOME, PNOME, ORE_SETTIMANA
    FROM   IMPIEGATO, PROGETTO, LAVORA_A
    WHERE CF =IMP AND PNUMERO = PROGETTO

UPDATE TABELLA_LAVORA_A_RIVISTA
SET       PNOME = 'PROGETTOB'
WHERE  COGNOME = 'ROSSI' AND NOME = 'MARIO' AND
          PNOME = 'PROGETTOA'
```

Quando una vista può essere modificata?

Standard SQL: una vista è aggiornabile solo quando una sola tupla di ciascuna tabella di base corrisponde ad una tupla della vista (ciò, ad esempio, implica che la vista non può coinvolgere funzioni aggregate).

I **sistemi commerciali** sono più restrittivi: una vista è aggiornabile solo se è definita su una sola tabella (alcuni sistemi chiedono anche che gli attributi della vista contengano la chiave primaria della tabella e che a tutti gli attributi della vista sia associato il vincolo NOT NULL ma non un valore di deafault).

La clausola CHECK OPTION può essere utilizzata solo nel contesto di tali viste: essa specifica che sono consentiti aggiornamenti solo delle tuple della vista e impone che dopo la loro esecuzione tali tuple continuino ad **appartenere alla vista** (una tupla può essere rimossa dalla vista se ad una degli attributi della vista viene assegnato un valore che rende falsa la condizione della clausola WHERE).

Esempio. Non è possibile operare una riduzione dello stipendio di un impiegato del Dipartimento 5 da un valore superiore a 10000 euro ad un valore minore o uguale a 10000 euro.

Opzione LOCAL o CASCADE

Nel caso una vista sia definita in termini di altre viste,

opzione LOCAL: il controllo sul mantenimento dell'appartenenza delle tuple alla vista viene fatto solo all'ultimo livello (la modifica non deve provocare una violazione della condizione che definisce la vista più esterna);

opzione CASCADED (opzione di default): il controllo sul mantenimento dell'appartenenza delle tuple alla vista viene fatto a tutti i livelli (si controlla che le tuple su cui si apportano le modifiche non scompaiano dalla vista per effetto della violazione di una qualsiasi delle condizioni delle viste coinvolte).

Esempi. Un assegnamento del valore 8000 euro ad una delle tuple della vista *IMPIEGATI5POVERI* è accettato con l'opzione LOCAL, ma è rifiutato con l'opzione CASCADED (default). Una modifica che assegna ad una tupla della vista il valore 40000 euro è rifiutato con entrambe le opzioni.