Interrogazioni nidificate

- Nella clausola `where` si possono utilizzare valori prodotti da altre istruzioni `select` utilizzando `any` (qualsiasi) o `all` (tutti) insieme agli operatori di confronto

Trovare nome, cognome e matricola degli studenti che non hanno fatto esami

```
select Matricola,Nome,Cognome from studenti
where matricola <> all (select studente
                        from esami
                        group by studente)
```

"diversa da tutte"
(si può usare anche **not in**)

Matricole degli studenti che hanno fatto almeno un esame

Un altra interrogazione

Trovare il nome degli studenti che hanno fatto almeno 3 esami

```
select Nome from studenti
where matricola = any (select studente
                       from esami
                       group by studente
                       having count(*)>=3)
```

"uguale ad almeno una"
(si può usare anche **in**)

Matricole degli studenti che hanno fatto almeno 3 esami

Interrogazioni nidificate semplici e complesse - 1

Studenti

Matricola	Cognome	Nome	Data di nascita
A80198760	Bianchi	Anna	22/03/1967
A80293450	Rossi	Andrea	13/04/1968
A80295640	Felici	Lorenzo	25/02/1969

Esami

Studente	Voto	Corso
A80198760	28	Analisi I
A80293450	30	Basi di Dati
A80295640	27	Analisi I
A80198760	30L	Fisica I
A80293450	21	Chimica

Interrogazioni nidificate semplici e complesse - 2

- Nei casi più complessi la sottointerrogazione deve essere eseguita una volta per ogni riga dell'interrogazione principale

Interrogazione semplice: cercare il cognome degli studenti con almeno un 30

```
select Cognome from studenti
where matricola =
           any (select studente from esami where voto=30)
```

Interrogazione complessa: cercare gli studenti con un voto uguale al giorno di nascita (c'è infatti da valutare la funzione "day" sulla data di nascita)

```
select Matricola from studenti
where matricola =
      any (select studente from esami where
            voto=day("data di nascita"))
```

Interrogazioni equivalenti

- Alcune interrogazioni possono essere espresse in diverse forme
- L'utente sceglie la forma più leggibile, il DBMS tenta di eseguire la forma più efficiente

Due interrogazioni equivalenti

```
select Matricola from studenti
where matricola = any (select studente from esami where voto=30)
```

```
select Matricola from studenti, esami
where matricola = studente and voto=30
```


Operatori su insiemi

- SQL mette a disposizione gli operatori `union` (unione), `except` (differenza), `intersect` (intersezione)

Cognomi e nomi di studenti che hanno sostenuto almeno un esame

```
select cognomi as nomeOppureCognome from studenti
where matricola = any (select studente from esami)

union

select nome as nomeOppureCognome from studenti
where matricola = any (select studente from esami)
```

Differenza

Cognomi che non sono anche nomi di studenti che hanno sostenuto almeno un esame

```
select cognomi as nomeOppureCognome from studenti
where matricola = any (select studente from esami)

except

select nome as nomeOppureCognome  from studenti
where matricola = any (select studente from esami)
```


Cancellazione e modifica di tuple

Cancellazione

Cancella lo studente con matricola A80198760

```
delete from studenti
where Matricola = 'A80198760'
```

Cancella gli studenti che non hanno esami

```
delete from studenti
where Matricola not in (select Matricola
                        from esami
                        group by Matricola)
```

Modifica

```
update esami set Voto = 30
where Matricola = 'A80198760'
      and corso = 'Analisi I'
```

Cancellazione: osservazioni

- Si eliminano tutti record indicati dalla clausola `where`
- Se la clausola where non è presente, si eliminano tutti i record della tabella

Rimuovere tutti gli studenti

```
delete from studenti
```

- Si può causare la cancellazione di record in altre tabelle (se i vincoli di integrità referenziale sono definiti con politiche di reazione **cascade**)

Modifica: osservazioni

- L'ordine dei comandi di aggiornamento è importante

Aumentare di 1 il voto degli studenti con voto maggiore di 25 e di 2 il voto degli altri studenti

```
update esami set voto=voto+2 where voto<=25

update esami set voto=voto+1 where voto>25
```

Chi aveva 25 dopo ha 28

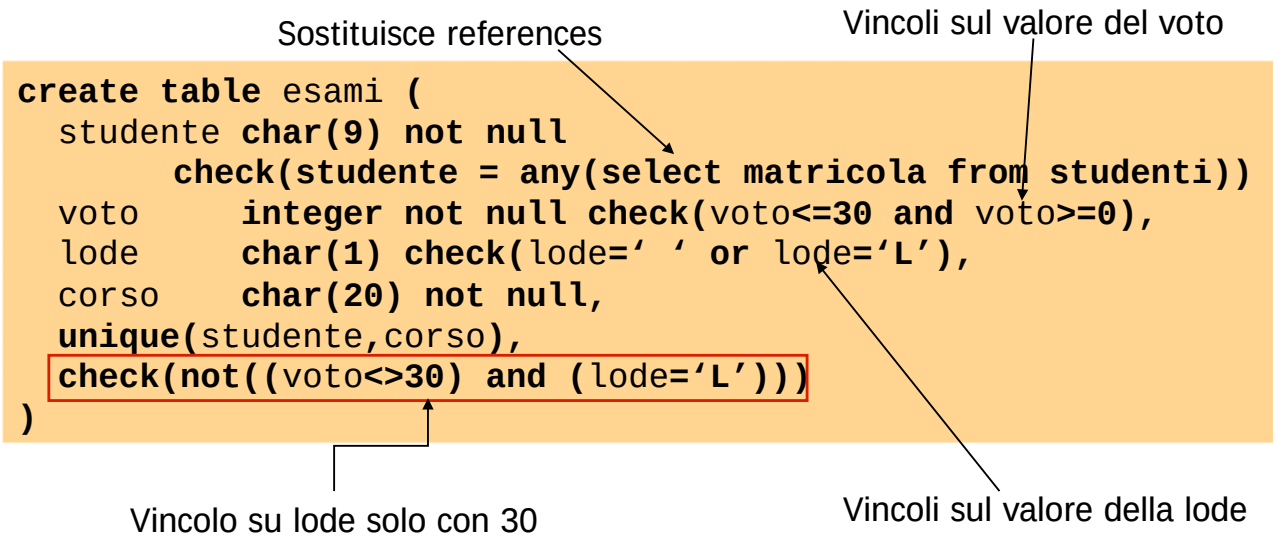
```
update esami set voto=voto+1 where voto>25

update esami set voto=voto+2 where voto<=25
```

Chi aveva 25 dopo ha 27

Vincoli di integrità generici: check

- E' possibile specificare vincoli di integrità generici con la clausola **check**(*condizione*)



Viste – 1/2

- Corrispondono a tabelle virtuali ovvero che non esistono fisicamente ma il cui contenuto è ottenuto da altre tabelle
- In SQL si ottengono assegnando una lista di attributi e un nome ad una interrogazione con **select**

```
create view StudentiMeritevoli
(Matricola,Nome,Cognome)
as select Matricola,Nome,Cognome
   from Studenti
   where Matricola in (select studente
                       from esami
                       group by studente
                       having avg(Voto)>=28)
```

Viste – 2/2

- E’ possibile fare delle modifiche attraverso la vista, purché ogni riga della vista corrisponda ad una sola riga della tabella

```
create view StudentiEsami
(Nome,Cognome,Corso,Voto,Matricola)
as select Nome,Cognome,Corso,Voto,Studente
   from Studenti, Esami
   where Matricola=Studente
```

Comando permesso

```
update StudentiEsami set voto=30
where cognome='rossi' and
corso='analisi'
```