

Atzeni, Ceri, Paraboschi, Torlone
Basi di dati
McGraw-Hill, 1996-2002

Capitolo 4:
SQL

24/09/2002

SQL

- originariamente "Structured Query Language", ora "nome proprio"
- linguaggio con varie funzionalità:
 - contiene sia il DDL sia il DML
- ne esistono varie versioni
- vediamo gli aspetti essenziali, non i dettagli

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

2

SQL: "storia"

- prima proposta SEQUEL (1974);
- prime implementazioni in SQL/DS e Oracle (1981)
- dal 1983 ca. "standard di fatto"
- standard (1986, poi 1989 e infine 1992, 1999)
 - recepito solo in parte (!!)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

3

Definizione dei dati in SQL

- Istruzione CREATE TABLE
 - definisce uno schema di relazione e ne crea un'istanza vuota
 - specifica attributi, domini e vincoli

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

4

CREATE TABLE, esempio

```
CREATE TABLE Impiegato(  
  Matricola CHAR(6) PRIMARY KEY,  
  Nome CHAR(20) NOT NULL,  
  Cognome CHAR(20) NOT NULL,  
  Dipart CHAR(15),  
  Stipendio NUMERIC(9) DEFAULT 0,  
  FOREIGN KEY(Dipart) REFERENCES  
    Dipartimento(NomeDip),  
  UNIQUE (Cognome, Nome)  
)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

5

Domini

- Domini elementari (predefiniti)
- Domini definiti dall'utente (semplici, ma riutilizzabili)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

6

Domini elementari

- **Carattere:** singoli caratteri o stringhe, anche di lunghezza variabile
- **Bit:** singoli booleani o stringhe
- **Numerici,** esatti e approssimati
- **Data, ora,** intervalli di tempo
- **Introdotti in SQL:1999:**
 - **Boolean**
 - **BLOB, CLOB** (binary/character large object): per grandi immagini e testi

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

7

Definizione di domini

- **Istruzione CREATE DOMAIN:**
 - **definisce un dominio (semplice), utilizzabile in definizioni di relazioni, anche con vincoli e valori di default**

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

8

CREATE DOMAIN, esempio

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

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

9

Vincoli intrarelazionali

- **NOT NULL**
- **UNIQUE** definisce chiavi
- **PRIMARY KEY:** chiave primaria (una sola, implica NOT NULL)
- **CHECK,** vedremo più avanti

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

10

UNIQUE e PRIMARY KEY

- **due forme:**
 - **nella definizione di un attributo, se forma da solo la chiave**
 - **come elemento separato**

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

11

CREATE TABLE, esempio

```
CREATE TABLE Impiegato(
  Matricola CHAR(6) PRIMARY KEY,
  Nome CHAR(20) NOT NULL,
  Cognome CHAR(20) NOT NULL,
  Dipart CHAR(15),
  Stipendio NUMERIC(9) DEFAULT 0,
  FOREIGN KEY( Dipart) REFERENCES
    Dipartimento(NomeDip),
  UNIQUE (Cognome,Nome)
)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

12

PRIMARY KEY, alternative

Matricola CHAR(6) PRIMARY KEY

Matricola CHAR(6),

....,

PRIMARY KEY (Matricola)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

13

CREATE TABLE, esempio

```
CREATE TABLE Impiegato(  
  Matricola CHAR(6) PRIMARY KEY,  
  Nome CHAR(20) NOT NULL,  
  Cognome CHAR(20) NOT NULL,  
  Dipart CHAR(15),  
  Stipendio NUMERIC(9) DEFAULT 0,  
  FOREIGN KEY(Dipart) REFERENCES  
    Dipartimento(NomeDip),  
  UNIQUE (Cognome, Nome)  
)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

14

Chiavi su più attributi, attenzione

Nome CHAR(20) NOT NULL,
Cognome CHAR(20) NOT NULL,
UNIQUE (Cognome, Nome),

Nome CHAR(20) NOT NULL UNIQUE,
Cognome CHAR(20) NOT NULL UNIQUE,

- Non è la stessa cosa!

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

15

Vincoli interrelazionali

- CHECK, vedremo più avanti
- REFERENCES e FOREIGN KEY permettono di definire vincoli di integrità referenziale
- di nuovo due sintassi
 - per singoli attributi
 - su più attributi
- E' possibile definire politiche di reazione alla violazione

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

16

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

Vigili	Matricola	Cognome	Nome
	3987	Rossi	Luca
	3295	Neri	Piero
	9345	Neri	Mario
	7543	Mori	Gino

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

17

Infrazioni

Codice	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

Auto	Prov	Numero	Cognome	Nome
	MI	39548K	Rossi	Mario
	TO	E39548	Rossi	Mario
	PR	839548	Neri	Luca

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

18

CREATE TABLE, esempio

```
CREATE TABLE Infrazioni(  
  Codice CHAR(6) NOT NULL PRIMARY KEY,  
  Data DATE NOT NULL,  
  Vigile INTEGER NOT NULL  
    REFERENCES Vigili(Matricola),  
  Provincia CHAR(2),  
  Numero CHAR(6) ,  
  FOREIGN KEY(Provincia, Numero)  
    REFERENCES Auto(Provincia, Numero)  
)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

19

Modifiche degli schemi

```
ALTER DOMAIN  
ALTER TABLE  
DROP DOMAIN  
DROP TABLE
```

...

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

20

Definizione degli indici

- è rilevante dal punto di vista delle prestazioni
- ma è a livello fisico e non logico
- in passato era importante perché in alcuni sistemi era l'unico mezzo per definire chiavi
- CREATE INDEX

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

21

DDL, in pratica

- In molti sistemi si utilizzano strumenti diversi dal codice SQL per definire lo schema della base di dati

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

22

SQL, operazioni sui dati

- interrogazione:
 - SELECT
- modifica:
 - INSERT, DELETE, UPDATE

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

23

Istruzione SELECT (versione base)

```
SELECT ListaAttributi  
FROM ListaTabelle  
[ WHERE Condizione ]
```

- "target list"
- clausola FROM
- clausola WHERE

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

24

Maternità			Persone		
Madre	Figlio		Nome	Età	Reddito
Luisa	Maria		Andrea	27	21
Luisa	Luigi		Aldo	25	15
Anna	Olga		Maria	55	42
Anna	Filippo		Anna	50	35
Maria	Andrea		Filippo	26	30
Maria	Aldo		Luigi	50	40
Paternità					
Padre	Figlio				
Sergio	Franco		Franco	60	20
Luigi	Olga		Olga	30	41
Luigi	Filippo		Sergio	85	35
Franco	Andrea		Luisa	75	87
Franco	Aldo				

24/09/2002 ischi-Torlone,
Basi di dati, Capitolo 4 25

Selezione e proiezione

- Nome e reddito delle persone con meno di trenta anni

```

PROJNome, Reddito(SELEta<30(Persone))

select nome, reddito
from persone
where eta < 30

```

24/09/2002 Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4 26

Persone	
Nome	Reddito
Andrea	21
Aldo	15
Filippo	30

24/09/2002 Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4 27

SELECT, abbreviazioni

```

select nome, reddito
from persone
where eta < 30

select p.nome as nome,
       p.reddito as reddito
from persone p
where p.eta < 30

```

24/09/2002 Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4 28

Selezione, senza proiezione

- Nome, età e reddito delle persone con meno di trenta anni

```

SELEta<30(Persone)

select *
from persone
where eta < 30

```

24/09/2002 Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4 29

SELECT, abbreviazioni

```

select *
from persone
where eta < 30

select nome, età, reddito
from persone
where eta < 30

```

24/09/2002 Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4 30

Proiezione, senza selezione

- Nome e reddito di tutte le persone

`PROJNome, Reddito(Persone)`

```
select nome, reddito
from persone
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

31

SELECT, abbreviazioni

- `R(A,B)`

```
select *
from R
```

equivale (intuitivamente) a

```
select X.A as A, X.B as B
from R X
where true
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

32

Espressioni nella target list

```
select Reddito/2 as redditoSemestrale
from Persone
where Nome = 'Luigi'
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

33

Condizione complessa

```
select *
from persone
where reddito > 25
and (eta < 30 or eta > 60)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

34

Condizione "LIKE"

- Le persone che hanno un nome che inizia per 'A' e ha una 'd' come terza lettera

```
select *
from persone
where nome like 'A_d%'
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

35

Gestione dei valori nulli

Impiegati

Matricola	Cognome	Filiale	Età
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

- Gli impiegati la cui età è o potrebbe essere maggiore di 40

```
SEL Età > 40 OR Età IS NULL (Impiegati)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

36

- Gli impiegati la cui età è o potrebbe essere maggiore di 40

SEL Età > 40 OR Età IS NULL (Impiegati)

```
select *
from impiegati
where eta > 40 or eta is null
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

37

Selezione, proiezione e join

- Istruzioni **SELECT** con una sola relazione nella clausola **FROM** permettono di realizzare:
 - selezioni, proiezioni, ridenominazioni
- con più relazioni nella **FROM** si realizzano **join** (e prodotti cartesiani)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

38

SQL e algebra relazionale

- R1(A1,A2) R2(A3,A4)

```
select R1.A1, R2.A4
from R1, R2
where R1.A2 = R2.A3
```

- prodotto cartesiano (**FROM**)
- selezione (**WHERE**)
- proiezione (**SELECT**)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

39

SQL e algebra relazionale, 2

- R1(A1,A2) R2(A3,A4)

```
select R1.A1, R2.A4
from R1, R2
where R1.A2 = R2.A3
```

$PROJ_{A1,A4}(SEL_{A2=A3}(R1 JOIN R2))$

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

40

- possono essere necessarie ridenominazioni
 - nel prodotto cartesiano
 - nella target list

```
select X.A1 AS B1, ...
from R1 X, R2 Y, R1 Z
where X.A2 = Y.A3 AND ...
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

41

```
select X.A1 AS B1, Y.A4 AS B2
from R1 X, R2 Y, R1 Z
where X.A2 = Y.A3 AND Y.A4 = Z.A1
```

$REN_{B1,B2 \rightarrow A1,A4}(PROJ_{A1,A4}(SEL_{A2=A3 AND A4=C1}(R1 JOIN R2 JOIN REN_{C1,C2 \rightarrow A1,A2}(R1))))$

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

42

SQL: esecuzione delle interrogazioni

- Le espressioni SQL sono dichiarative e noi ne stiamo vedendo la semantica
- In pratica, i DBMS eseguono le operazioni in modo efficiente, ad esempio:
 - eseguono le selezioni al più presto
 - se possibile, eseguono join e non prodotti cartesiani

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

43

SQL: specificità delle interrogazioni

- La capacità dei DBMS di "ottimizzare" le interrogazioni, rende (di solito) non necessario preoccuparsi dell'efficienza quando si specifica un'interrogazione
- È perciò più importante preoccuparsi della chiarezza (anche perché così è più difficile sbagliare ...)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

44

Proiezione, attenzione

- cognome e filiale di tutti gli impiegati

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma

PROJ_{Cognome, Filiale} (Impiegati)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

45

```
select
  cognome, filiale
from impiegati
```

```
select distinct
  cognome, filiale
from impiegati
```

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma
Rossi	Roma

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

46

Maternità	Madre	Figlio	Persone		
			Nome	Età	Reddito
	Luisa	Maria	Andrea	27	21
	Luisa	Luigi	Aldo	25	15
	Anna	Olga	Maria	55	42
	Anna	Filippo	Anna	50	35
	Maria	Andrea	Filippo	26	30
	Maria	Aldo	Luigi	50	40
			Franco	60	20
	Sergio	Franco	Olga	30	41
	Luigi	Olga	Sergio	85	35
	Luigi	Filippo	Luisa	75	87
	Franco	Andrea			
	Franco	Aldo			

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

47

Selezione, proiezione e join

- I padri di persone che guadagnano più di venti milioni

```
PROJPadre(paternita
  JOINFiglio = Nome
  SELReddito > 20(persone))
```

```
select distinct padre
from persone, paternita
where figlio = nome and reddito > 20
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

48

Join naturale

- Padre e madre di ogni persona

paternita JOIN maternita

```
select paternita.figlio,padre, madre
from maternita, paternita
where paternita.figlio = maternita.figlio
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

49

- Le persone che guadagnano più dei rispettivi padri; mostrare nome, reddito e reddito del padre

```
PROJNome, Reddito, RP (SELReddito>RP
(RENNP,EP,RP < Nome, Eta, Reddito(persone)
JOINNP=Padre
(paternita JOINFiglio =Nome persone)))
```

```
select f.nome, f.reddito, p.reddito
from persone p, paternita, persone f
where p.nome = padre and
figlio = f.nome and
f.reddito > p.reddito
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

50

SELECT, con ridenominazione del risultato

```
select figlio, f.reddito as reddito,
p.reddito as redditoPadre
from persone p, paternita, persone f
where p.nome = padre and figlio = f.nome
and .reddito > p.reddito
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

51

Join esplicito

- Padre e madre di ogni persona

```
select paternita.figlio,padre, madre
from maternita, paternita
where paternita.figlio = maternita.figlio
```

```
select madre, paternita.figlio, padre
from maternita join paternita on
paternita.figlio = maternita.figlio
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

52

SELECT con join esplicito, sintassi

```
SELECT ...
FROM Tabella { ... JOIN Tabella ON CondDiJoin }, ...
[ WHERE AltraCondizione ]
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

53

- Le persone che guadagnano più dei rispettivi padri; mostrare nome, reddito e reddito del padre

```
select f.nome, f.reddito, p.reddito
from persone p, paternita, persone f
where p.nome = padre and
figlio = f.nome and
f.reddito > p.reddito
```

```
select f.nome, f.reddito, p.reddito
from persone p join paternita on p.nome = padre
join persone f on figlio = f.nome
where f.reddito > p.reddito
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

54

Ulteriore estensione: join naturale (meno diffuso)

PROJ_{Figlio,Padre,Madre}(
 paternita JOIN_{Figlio = Nome} REN_{Nome=Figlio}(maternita))

paternita JOIN maternita

```
select madre, paternita.figlio, padre
from maternita join paternita on
paternita.figlio = maternita.figlio
```

```
select madre, paternita.figlio, padre
from maternita natural join paternita
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

55

Join esterno: "outer join"

- Padre e, se nota, madre di ogni persona

```
select paternita.figlio, padre, madre
from paternita left join maternita
on paternita.figlio = maternita.figlio
```

```
select paternita.figlio, padre, madre
from paternita left outer join maternita
on paternita.figlio = maternita.figlio
```

- outer e' opzionale

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

56

Outer join

```
select paternita.figlio, padre, madre
from maternita join paternita
on maternita.figlio = paternita.figlio
```

```
select paternita.figlio, padre, madre
from maternita left outer join paternita
on maternita.figlio = paternita.figlio
```

```
select paternita.figlio, padre, madre
from maternita full outer join paternita
on maternita.figlio = paternita.figlio
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

57

Ordinamento del risultato

- Nome e reddito delle persone con meno di trenta anni in ordine alfabetico

```
select nome, reddito
from persone
where eta < 30
order by nome
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

58

```
select nome, reddito
from persone
where eta < 30
```

Persone	
Nome	Reddito
Andrea	21
Aldo	15
Filippo	30

```
select nome, reddito
from persone
where eta < 30
order by nome
```

Persone	
Nome	Reddito
Aldo	15
Andrea	21
Filippo	30

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

59

Operatori aggregati

- Nelle espressioni della target list possiamo avere anche espressioni che calcolano valori a partire da insiemi di ennuple:

- conteggio, minimo, massimo, media, totale
- sintassi base (semplificata):

Funzione ([DISTINCT] *)
 Funzione ([DISTINCT] Attributo)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

60

Operatori aggregati: COUNT

- Il numero di figli di Franco

```
select count(*) as NumFigliDiFranco
from Paternita
where Padre = 'Franco'
```

- l'operatore aggregato (count) viene applicato al risultato dell'interrogazione:

```
select *
from Paternita
where Padre = 'Franco'
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

61

Paternità	Padre	Figlio
	Sergio	Franco
	Luigi	Olga
	Luigi	Filippo
	Franco	Andrea
	Franco	Aldo

```
NumFigliDiFranco
2
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

62

COUNT e valori nulli

```
select count(*) from persone
```

```
select count(reddito) from persone
```

```
select count(distinctreddito) from persone
```

Persone	Nome	Età	Reddito
	Andrea	27	21
	Aldo	25	NULL
	Maria	55	21
	Anna	50	35

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

63

Altri operatori aggregati

- SUM, AVG, MAX, MIN
- Media dei redditi dei figli di Franco

```
select avg(reddito)
from persone join paternita on nome=figlio
where padre='Franco'
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

64

Operatori aggregati e valori nulli

```
select avg(reddito) as redditomedio
from persone
```

Persone	Nome	Età	Reddito
	Andrea	27	30
	Aldo	25	NULL
	Maria	55	36
	Anna	50	36

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

65

Operatori aggregati e target list

- un'interrogazione scorretta:

```
select nome, max(reddito)
from persone
```

- di chi sarebbe il nome? La target list deve essere omogenea

```
select min(eta), avg(reddito)
from persone
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

66

Operatori aggregati e raggruppamenti

- Le funzioni possono essere applicate a partizioni delle relazioni
- Clausola GROUP BY:
GROUP BY listaAttributi

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

67

Operatori aggregati e raggruppamenti

- Il numero di figli di ciascun padre

```
select padre, count(*) AS NumFigli
from paternita
group by Padre
```

paternita	Padre	Figlio	Padre	NumFigli
	Sergio	Franco	Sergio	1
	Luigi	Olga	Luigi	2
	Luigi	Filippo	Luigi	2
	Franco	Andrea	Franco	2
	Franco	Aldo	Franco	2

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

68

Semantica di interrogazioni con operatori aggregati e raggruppamenti

1. interrogazione senza group by e senza operatori aggregati
select *
from paternita
2. si raggruppa e si applica l'operatore aggregato a ciascun gruppo

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

69

Raggruppamenti e target list

scorretta

```
select padre, avg(f.reddito), p.reddito
from persone f join paternita on figlio = nome join
persone p on padre =p.nome
group by padre
```

corretta

```
select padre, avg(f.reddito), p.reddito
from persone f join paternita on figlio = nome join
persone p on padre =p.nome
group by padre, p.reddito
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

70

Condizioni sui gruppi

- I padri i cui figli hanno un reddito medio maggiore di 25

```
select padre, avg(f.reddito)
from persone f join paternita on figlio = nome
group by padre
having avg(f.reddito) > 25
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

71

WHERE o HAVING?

- I padri i cui figli sotto i 30 anni hanno un reddito medio maggiore di 20

```
select padre, avg(f.reddito)
from persone f join paternita on figlio = nome
where eta < 30
group by padre
having avg(f.reddito) > 25
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

72

Sintassi, riassumiamo

```
SelectSQL ::=
select ListaAttributiOEspressioni
from ListaTabelle
[ where CondizioniSemplici ]
[ group by
  ListaAttributiDiRaggruppamento ]
[ having CondizioniAggregate ]
[ order by ListaAttributiDiOrdinamento ]
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

73

Unione, intersezione e differenza

- La select da sola non permette di fare unioni; serve un costrutto esplicito:

```
select ...
union [all]
select ...
```

- i duplicati vengono eliminati (a meno che si usi all); anche dalle proiezioni!

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

74

Notazione posizionale!

```
select padre
from paternita
union
select madre
from maternita
```

- quali nomi per gli attributi del risultato?
 - nessuno
 - quelli del primo operando
 - ...

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

75

Figlio		Padre	Figlio
Sergio	Franco	Sergio	Franco
Luigi	Olga	Luigi	Olga
Luigi	Filippo	Luigi	Filippo
Franco	Andrea	Franco	Andrea
Franco	Aldo	Franco	Aldo
Luisa	Maria	Luisa	Maria
Luisa	Luigi	Luisa	Luigi
Anna	Olga	Anna	Olga
Anna	Filippo	Anna	Filippo
Maria	Andrea	Maria	Andrea
Maria	Aldo	Maria	Aldo

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

76

Notazione posizionale, 2

```
select padre, figlio      select padre, figlio
from paternita           from paternita
union                    union
select figlio, madre     select madre, figlio
from maternita           from maternita
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

77

Notazione posizionale, 3

- Anche con le ridenominazioni non cambia niente:

```
select padre as genitore, figlio
from paternita
union
select figlio, madre as genitore
from maternita
```
- Corretta:

```
select padre as genitore, figlio
from paternita
union
select madre as genitore, figlio
from maternita
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

78

Differenza

```
select Nome
from Impiegato
except
select Cognome as Nome
from Impiegato
```

- vedremo che si può esprimere con select nidificate

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

79

Intersezione

```
select Nome
from Impiegato
intersect
select Cognome as Nome
from Impiegato
```

- equivale a

```
select I.Nome
from Impiegato I, Impiegato J
where I.Nome = J.Cognome
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

80

Interrogazioni nidificate

- le condizioni atomiche permettono anche
 - il confronto fra un attributo (o più, vedremo poi) e il risultato di una sottointerrogazione
 - quantificazioni esistenziali

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

81

- nome e reddito del padre di Franco

```
select Nome, Reddito
from Persone, Paternita
where Nome = Padre and Figlio = 'Franco'
```

```
select Nome, Reddito
from Persone
where Nome = ( select Padre
                from Paternita
                where Figlio = 'Franco')
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

82

Interrogazioni nidificate, commenti

- La forma nidificata è “meno dichiarativa”, ma talvolta più leggibile (richiede meno variabili)
- La forma piana e quella nidificata possono essere combinate
- Le sottointerrogazioni non possono contenere operatori insiemistici (“l’unione si fa solo al livello esterno”); la limitazione non è significativa

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

83

- Nome e reddito dei padri di persone che guadagnano più di 20 milioni

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita , Persone F
where P.Nome = Padre and Figlio = F.Nome
and F.Reddito > 20
```

```
select Nome, Reddito
from Persone
where Nome in (select Padre
                from Paternita
                where Figlio = any (select Nome
                                    from Persone
                                    where Reddito > 20))
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

84

- Nome e reddito dei padri di persone che guadagnano più di 20 milioni

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
and F.Reddito > 20
```

```
select Nome, Reddito
from Persone
where Nome in (select Padre
from Paternita, Persone
where Figlio = Nome
and Reddito > 20)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

85

Interrogazioni nidificate, commenti, 2

- La prima versione di SQL prevedeva solo la forma nidificata (o strutturata), con una sola relazione in ogni clausola FROM. Insoddisfacente:
 - la dichiaratività è limitata
 - non si possono includere nella target list attributi di relazioni nei blocchi interni

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

86

- Nome e reddito dei padri di persone che guadagnano più di 20 milioni, con indicazione del reddito del figlio

```
select distinct P.Nome, P.Reddito, F.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
and F.Reddito > 20
```

```
select Nome, Reddito, ????
from Persone
where Nome in (select Padre
from Paternita
where Figlio = any (select Nome
from Persone
where Reddito > 20))
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

87

Interrogazioni nidificate, commenti, 3

- regole di visibilità:
 - non è possibile fare riferimenti a variabili definite in blocchi più interni
 - se un nome di variabile è omissso, si assume riferimento alla variabile più "vicina"
 - in un blocco si può fare riferimento a variabili definite in blocchi più esterni; la semantica base (prodotto cartesiano, selezione, proiezione) non funziona più, vedremo presto

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

88

Quantificazione esistenziale

- Ulteriore tipo di condizione
 - EXISTS (Sottoespressione)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

89

- Le persone che hanno almeno un figlio

```
select *
from Persone
where exists (
  select *
  from Paternita
  where Padre = Nome) or
exists (
  select *
  from Maternita
  where Madre = Nome)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

90

- I padri i cui figli guadagnano tutti più di venti milioni

```
select distinct Padre
from Paternita Z
where not exists (
  select *
  from Paternita W, Persone
  where W.Padre = Z.Padre
  and W.Figlio = Nome
  and Reddito <= 20)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

91

Semantica delle espressioni “correlate”

- L'interrogazione interna viene eseguita una volta per ciascuna ennupla dell'interrogazione esterna

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

92

Visibilità

- scorretta:

```
select *
from Impiegato
where Dipart in (select Nome
  from Dipartimento D1
  where Nome = 'Produzione') or
  Dipart in (select Nome
  from Dipartimento D2
  where D2.Citta = D1.Citta)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

93

Disgiunzione e unione (ma non sempre)

```
select * from Persone where Reddito > 30
union
select F.*
from Persone F, Paternita, Persone P
where F.Nome = Figlio and Padre = P.Nome
and P.Reddito > 30

select *
from Persone F
where Reddito > 30 or
exists (select *
  from Paternita, Persone P
  where F.Nome = Figlio and Padre = P.Nome
  and P.Reddito > 30)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

94

Differenza e nidificazione

```
select Nome from Impiegato
except
select Cognome as Nome from Impiegato

select Nome
from Impiegato I
where not exists (select *
  from Impiegato
  where Cognome = I.Nome)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

95

Massimo e nidificazione

- La persona (o le persone) con il reddito massimo

```
select *
from persone
where reddito = ( select max(reddito)
  from persone)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

96

Operazioni di aggiornamento

- operazioni di
 - inserimento: insert
 - eliminazione: delete
 - modifica: update
- di una o più ennuple di una relazione
- sulla base di una condizione che può coinvolgere anche altre relazioni

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

97

Inserimento

```
INSERT INTO Tabella [ ( Attributi ) ]  
VALUES( Valori )
```

oppure

```
INSERT INTO Tabella [ ( Attributi )]  
SELECT ...
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

98

```
INSERT INTO Persone VALUES ('Mario',25,52)
```

```
INSERT INTO Persone(Nome, Eta, Reddito)  
VALUES('Pino',25,52)
```

```
INSERT INTO Persone(Nome, Reddito)  
VALUES('Lino',55)
```

```
INSERT INTO Persone ( Nome )  
SELECT Padre  
FROM Paternita  
WHERE Padre NOT IN (SELECT Nome  
FROM Persone)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

99

Inserimento , commenti

- l'ordinamento degli attributi (se presente) e dei valori è significativo
- le due liste debbono avere lo stesso numero di elementi
- se la lista di attributi è omessa, si fa riferimento a tutti gli attributi della relazione, secondo l'ordine con cui sono stati definiti
- se la lista di attributi non contiene tutti gli attributi della relazione, per gli altri viene inserito un valore nullo (che deve essere permesso) o un valore di default

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

100

Eliminazione di ennuple

```
DELETE FROM Tabella  
[ WHERE Condizione ]
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

101

```
DELETE FROM Persone  
WHERE Eta < 35
```

```
DELETE FROM Paternita  
WHERE Figlio NOT in ( SELECT Nome  
FROM Persone)
```

```
DELETE FROM Paternita
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

102

Eliminazione, commenti

- elimina le ennupe che soddisfano la condizione
- può causare (se i vincoli di integrità referenziale sono definiti con politiche di reazione cascade) eliminazioni da altre relazioni
- ricordare: se la where viene omessa, si intende where true

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

103

Modifica di ennupe

```
UPDATE NomeTabella
SET Attributo = < Espressione |
      SELECT ... |
      NULL |
      DEFAULT >
[ WHERE Condizione ]
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

104

```
UPDATE Persone SET Reddito = 45
WHERE Nome = 'Piero'
```

```
UPDATE Persone
SET Reddito = Reddito * 1.1
WHERE Eta < 30
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

105

Vincoli di integrità generici: check

- Specifica di vincoli di ennupe (e anche vincoli più complessi)
check (Condizione)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

106

Check, esempio

```
create table Impiegato
(
  Matricola character(6),
  Cognome character(20),
  Nome character(20),
  Sesso character not null check (sesso in ('M','F'))
  Stipendio integer,
  Superiore character(6),
  check (Stipendio <= (select Stipendio
                       from Impiegato J
                       where Superiore = J.Matricola)
)
)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

107

Vincoli di integrità generici: asserzioni

- Specifica vincoli a livello di schema
create assertion NomeAss check (Condizione)

```
create assertion AlmenoUnImpiegato
check (1 <= ( select count(*)
              from Impiegato ))
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

108

Viste

```
create view NomeVista [ ( ListaAttributi ) ] as SelectSQL  
[ with [ local | cascaded ] check option ]
```

```
create view ImpiegatiAmmin  
(Matricola, Nome, Cognome, Stipendio) as  
select Matricola, Nome, Cognome, Stipendio  
from Impiegato  
where Dipart = 'Amministrazione' and  
Stipendio > 10
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

109

Aggiornamenti sulle viste

- Ammessi (di solito) solo su viste definite su una sola relazione
- Alcune verifiche possono essere imposte

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

110

```
create view ImpiegatiAmminPoveri as  
select *  
from ImpiegatiAmmin  
where Stipendio < 50  
with check option
```

- check option permette modifiche, ma solo a condizione che la ennupla continui ad appartenere alla vista (non posso modificare lo stipendio portandolo a 60)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

111

Un'interrogazione non standard

- La nidificazione nella having non è ammessa

```
select Dipart  
from Impiegato  
group by Dipart  
having sum(Stipendio) >= all  
(select sum(Stipendio)  
from Impiegato  
group by Dipart)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

112

Soluzione con le viste

```
create view BudgetStipendi(Dip,TotaleStipendi) as  
select Dipart, sum(Stipendio)  
from Impiegato  
group by Dipart
```

```
select Dip  
from BudgetStipendi  
where TotaleStipendi =(select max(TotaleStipendi)  
from BudgetStipendi)
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

113

Ancora sulle viste

- Interrogazione scorretta
select avg(count(distinct Ufficio))
from Impiegato
group by Dipart
- Con una vista
create view DipartUffici(NomeDip,NroUffici) as
select Dipart, count(distinct Ufficio)
from Impiegato
group by Dipart;
select avg(NroUffici)
from DipartUffici

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

114

Viste ricorsive

- Per ogni impiegato, trovare tutti i superiori, avendo Supervisione (Impiegato, Capo)
- Serve la ricorsione ; in Datalog:

Superiore (Impiegato: i, SuperCapo: c) →
Supervisione (Impiegato: i, Capo: c)

Superiore (Impiegato: i, SuperCapo: c) →
Supervisione (Impiegato: i, Capo: c'),
Superiore (Impiegato: c', SuperCapo: c)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

115

Viste ricorsive in SQL:1999

```
with recursive Superiore(Imp,Supercapo
( ( select Imp, Capo as Supercapo
from Supervisione)
union
( select Superiore.Imp, Supercapo
from Supervisione, Superiore
where Supervisione.Capo = Superiore.Imp) )
select *
from Superiore
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

116

Controllo dell'accesso

- In SQL è possibile specificare chi (utente) e come (lettura, scrittura, ...) può utilizzare la base di dati (o parte di essa)
- Oggetto dei privilegi (diritti di accesso) sono di solito le tabelle, ma anche altri tipi di risorse, quali singoli attributi, viste o domini
- Un utente predefinito `_system` (amministratore della base di dati) ha tutti i privilegi
- Il creatore di una risorsa ha tutti i privilegi su di essa

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

117

Privilegi

- Un privilegio è caratterizzato da:
 - la risorsa cui si riferisce
 - l'utente che concede il privilegio
 - l'utente che riceve il privilegio
 - l'azione che viene permessa
 - la trasmissibilità del privilegio

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

118

Tipi di privilegi offerti da SQL

- `insert`: permette di inserire nuovi oggetti (ennuple)
- `update`: permette di modificare il contenuto
- `delete`: permette di eliminare oggetti
- `select`: permette di leggere la risorsa
- `references`: permette la definizione di vincoli di integrità referenziale verso la risorsa (può limitare la possibilità di modificare la risorsa)
- `usage`: permette l'utilizzo in una definizione (per esempio, di un dominio)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

119

grant e revoke

- Concessione di privilegi:
`grant < Privileges | all privileges > on Resource to Users [with grant option]`
 - `grant option` specifica se il privilegio può essere trasmesso ad altri utenti
`grant select on Department to Stefano`
- Revoca di privilegi
`revoke Privileges on Resource from Users [restrict | cascade]`

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

120

Transazione

- Insieme di operazioni da considerare indivisibile ("atomico"), corretto anche in presenza di concorrenza e con effetti definitivi
- Proprietà ("acide"):
 - Atomicità
 - Consistenza
 - Isolamento
 - Durabilità (persistenza)

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

121

Le transazioni sono ... atomiche

- La sequenza di operazioni sulla base di dati viene eseguita per intero o per niente:
 - trasferimento di fondi da un conto A ad un conto B: o si fanno il prelievo da A e il versamento su B o nessuno dei due

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

122



Le transazioni sono ... consistenti

- Al termine dell'esecuzione di una transazione, i vincoli di integrità debbono essere soddisfatti
- "Durante" l'esecuzione ci possono essere violazioni, ma se restano alla fine allora la transazione deve essere annullata per intero ("abortita")

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

123

Le transazioni sono ... isolate

- L'effetto di transazioni concorrenti deve essere coerente (ad esempio "equivalente" all'esecuzione separata)
 - se due assegni emessi sullo stesso conto corrente vengono incassati contemporaneamente si deve evitare di trascurarne uno

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

124



I risultati delle transazioni sono durevoli

- La conclusione positiva di una transazione corrisponde ad un impegno (in inglese commit) a mantenere traccia del risultato in modo definitivo, anche in presenza di guasti e di esecuzione concorrente

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

125

Transazioni in SQL

- Istruzioni fondamentali
 - begin transaction: specifica l'inizio della transazione (le operazioni non vengono eseguite sulla base di dati)
 - commit work: le operazioni specificate a partire dal begin transaction vengono eseguite
 - rollback work: si rinuncia all'esecuzione delle operazioni specificate dopo l'ultimo begin transaction

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

126

Una transazione in SQL

```
begin transaction;  
update ContoCorrente  
  set Saldo = Saldo - 10  
  where NumeroConto = 12345 ;  
update ContoCorrente  
  set Saldo = Saldo + 10  
  where NumeroConto = 55555 ;  
commit work;
```

24/09/2002

Atzeni-Ceri-Paraboschi-Torlone,
Basi di dati, Capitolo 4

127