SQL: ALTRE FORME DI JOIN, FUNZIONI AGGREGATIVE, ESPRESSIONI

Patrizio Dazzi
a.a. 2017 - 2018

COMUNICAZIONI

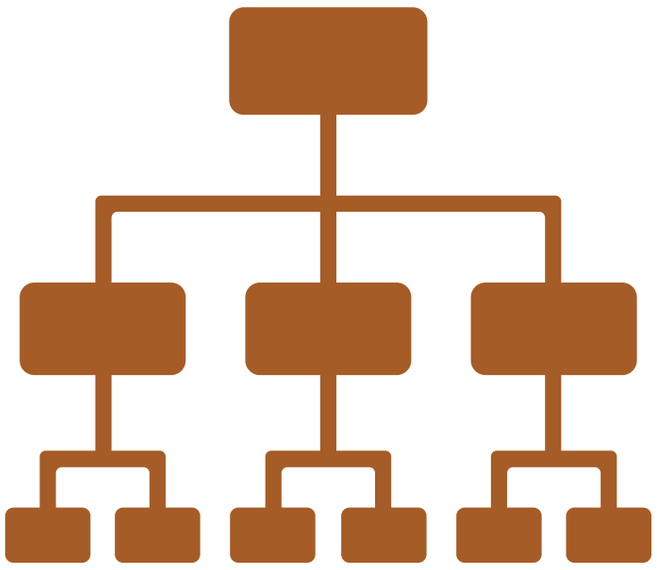
- Homeworks
  - preso il prossimo!

- Compitini
  - un po’ di pazienza…

- Assistente
  - Vinicius Monteiro De Lira

PICCOLO RIASSUNTO DELLA PUNTATA PRECEDENTE

- Siamo in grado di fare interrogazioni complesse
- Query annidate, quantificazioni, join multipli, raggruppamenti, ordinamenti…
- Dobbiamo passare alla progettazione
  - priva vediamo alcuni piccoli dettagli



ALTRE FORME DI JOIN

RICHIAMI: OPERATORI BINARI

- **Prodotto Cartesiano**
  - operatore binario R X S
  - correla le ennuple di R con quelle di S in tutti i possibili modi
  - il risultato ha tutti gli attributi di S e tutti gli attributi di R
  - $|R \times S| = |R| * |S|$
  - **prestazioni scadenti**

RICHIAMI: OPERATORI BINARI

- Join
  - operatore binario: $R \bowtie_{condizione} S$
  - condizione: uguaglianza tra attributi di R e S dello stesso tipo; es: R.a = S.b
  - il risultato è l'insieme di ennuple ottenute "concatenando" le ennuple di R con quelle di S, purché l'ennupla risultante soddisfi la condizione
  - consente interrogazioni efficienti

RICHIAMI: OPERATORI BINARI

- Nota
  - le condizioni di join tra le tabelle corrispondono normalmente ai vincoli di riferimento (es: studente ed esame)
  - ma non necessariamente è sempre così
  - es: join tra Professori e Studenti su Professori.nome=Studenti.nome
  - es: join tra Studenti e Corsi su Studenti.cognome=Corsi.titolo

ALTRE FORME DI JOIN

- Ci sono numerose varianti del Join
- **Join Naturale (ne avevamo già parlato!)**
  - variante del join senza condizione
  - condizione implicita: valori uguali sugli attributi con lo stesso nome
- **Poco utilizzato**
- Più utilizzati sono i join esterni

CARDINALITÀ DEL JOIN

- In un join ci possono essere ennuple di R e di S che non contribuiscono al risultato
- Join Completo
  - tutte le ennuple di R e le ennuple di S contribuiscono ad almeno un'ennupla del risultato
- Join Non Completo
  - non tutte le ennuple partecipano al risultato
  - Join Vuoto: caso degenere

JOIN COMPLETO

Professori

cod	cognome	nome	qualifica	facolta
FT	Totti	Francesco	ordinario	Ingegneria
VC	Vieri	Christian	associato	Scienze
ADP	Del Piero	Alessandro	supplente	null

Numeri

professore	numero
FT	0971205145
FT	347123456
VC	0971205227
ADP	0971205363
ADP	338123456

Professori $\bowtie_{\text{cod=professore}}$ Numeri

cod	cognome	nome	qualifica	facolta	professore	numero
FT	Totti	Francesco	ordinario	Ingegneria	FT	0971205145
FT	Totti	Francesco	ordinario	Ingegneria	FT	347123456
VC	Vieri	Christian	associato	Scienze	VC	0971205227
ADP	Del Piero	Alessandro	supplente	null	ADP	0971205363
ADP	Del Piero	Alessandro	supplente	null	ADP	338123456

JOIN COMPLETO E CARDINALITÀ

StudentiLaureaTriennale

matr	cognome	nome	ciclo	…
111	Rossi	Mario	I. tr.	…
222	Neri	Paolo	I. tr.	…
333	Rossi	Maria	I. tr.	…
444	Pinco	Palla	I. tr.	…

CorsiLaureaTriennale

cod	titolo	ciclo	…
PR1	Programmazione I	I. tr.	…
ASD	Algoritmi e Str. Dati	I. tr.	…

CorsiLaureaTriennale $\bowtie_{C.ciclo = S.ciclo}$ StudentiLaureaTriennale

cod	titolo	C.ciclo	…	matr	cognome	nome	S.ciclo	…
PR1	Programmazione I	I. tr.	…	111	Rossi	Mario	I. tr.	…
PR1	Programmazione I	I. tr.	…	222	Neri	Paolo	I. tr.	…
PR1	Programmazione I	I. tr.	…	333	Rossi	Maria	I. tr.	…
PR1	Programmazione I	I. tr.	…	444	Pinco	Palla	I. tr.	…
ASD	Algoritmi e Str. Dati	I. tr.	…	111	Rossi	Mario	I. tr.	…
ASD	Algoritmi e Str. Dati	I. tr.	…	222	Neri	Paolo	I. tr.	…
ASD	Algoritmi e Str. Dati	I. tr.	…	333	Rossi	Maria	I. tr.	…
ASD	Algoritmi e Str. Dati	I. tr.	…	444	Pinco	Palla	I. tr.	…

JOIN NON COMPLETO

Professori

cod	cognome	nome	qualifica	facolta
FT	Totti	Francesco	ordinario	Ingegneria
VC	Vieri	Christian	associato	Scienze
ADP	Del Piero	Alessandro	supplente	null

Numeri

professore	numero
FT	0971205145
FT	347123456
VC	0971205227
ADP	0971205363
ADP	338123456

Professori $\bowtie_{\text{cod=professore}}$ Numeri

supponendo che la ennupla sia eliminata

cod	cognome	nome	qualifica	facolta	professore	numero
FT	Totti	Francesco	ordinario	Ingegneria	FT	0971205145
FT	Totti	Francesco	ordinario	Ingegneria	FT	347123456
ADP	Del Piero	Alessandro	supplente	null	ADP	0971205363
ADP	Del Piero	Alessandro	supplente	null	ADP	338123456

JOIN VUOTO

Professori

cod	cognome	nome	…
FT	Totti	Francesco	…
VC	Vieri	Christian	…
ADP	Del Piero	Alessandro	…

Studenti

matr	cognome	nome	…
111	Rossi	Mario	…
222	Neri	Paolo	…
333	Rossi	Maria	…
444	Pinco	Palla	…
77777	Bruno	Pasquale	…
88888	Pinco	Pietro	…

Professori $\bowtie$ Professori.nome = Studenti.nome Studenti

matr	S.cognome	S.nome	ciclo	anno	relatore	P.cognome	P.nome	qualifica	facolta

CARDINALITÀ DEL JOIN

- Cardinalità del join di R_1 e R_2
  - compreso fra zero e $|R_1| * |R_2|$
- Caso tipico:
  - join tra la chiave primaria di R_1 e un attributo di R_2 su cui c'è un vincolo di integrità referenziale (chiave esterna)
  - cardinalità pari a $|R_2|$
  - es: Prof. e Numeri, con Prof.cod=Numeri.professore

JOIN ESTERNI

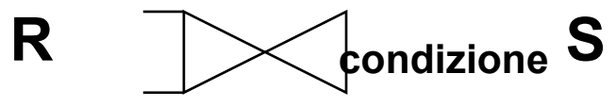
- Nei join incompleti si “perdono” ennuple
  - es: professori senza numeri di telefono

- In alcuni casi può essere un problema

- **Join Esterno (“Outer Join”)**
  - tutte le ennuple contribuiscono (completo)
  - le ennuple per cui non c’è controparte sono completate con valori nulli

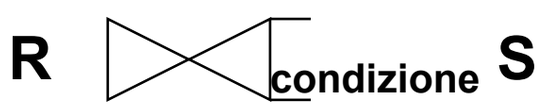
JOIN ESTERNI

- Join Esterno Sinistro



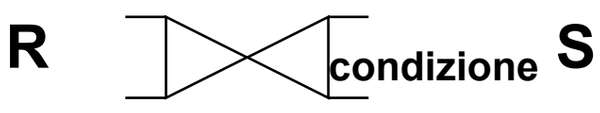
  - garantisce che vengono mantenute tutte le ennuple della tabella a sinistra (R)

- Join Esterno Destro



  - garantisce che vengono mantenute tutte le ennuple della tabella a destra (S)

- Join Esterno Completo



JOIN ESTERNO SINISTRO

Professori

cod	cognome	nome	qualifica	facolta
FT	Totti	Francesco	ordinario	Ingegneria
VC	Vieri	Christian	associato	Scienze
ADP	Del Piero	Alessandro	supplente	null

Numeri

professore	numero
FT	0971205145
FT	347123456
ADP	0971205363
ADP	338123456

Professori $\bowtie$ cod=professore Numeri

cod	cognome	nome	qualifica	facolta	professore	numero
FT	Totti	Francesco	ordinario	Ingegneria	FT	0971205145
FT	Totti	Francesco	ordinario	Ingegneria	FT	347123456
VC	Vieri	Christian	associato	Scienze	null	null
ADP	Del Piero	Alessandro	supplente	null	ADP	0971205363
ADP	Del Piero	Alessandro	supplente	null	ADP	338123456

JOIN ESTERNO COMPLETO

Professori

cod	cognome	nome	…
FT	Totti	Francesco	…
VC	Vieri	Christian	…
ADP	Del Piero	Alessandro	…

Studenti

matr	cognome	nome	…
111	Rossi	Mario	…
222	Neri	Paolo	…
333	Rossi	Maria	…
444	Pinco	Palla	…
77777	Bruno	Pasquale	…
88888	Pinco	Pietro	…

Professori ⟗ Professori.nome = Studenti.nome Studenti

cod	P.cognome	P.nome	...	matr	S.cognome	S.nome	...
FT	Totti	Francesco	...	null	null	null	null
null	null	null	null	111	Rossi	Mario	...
...	...	...	...	...	...	...	...

ALTRE FORME DI JOIN: SQL

- `R NATURAL JOIN S`

- `R LEFT [OUTER] JOIN S ON A=B`

- `R RIGHT [OUTER] JOIN S ON A=B`

- `R FULL [OUTER] JOIN S ON A=B`

FUNZIONI AGGREGATIVE : RICHIAMO

RICHIAMO: FUNZIONI AGGREGATIVE

- Visto che non sono state comprese troppo bene…
- In alcuni casi è utile avere valori calcolati
  - somme e medie di attributi numerici;
    es: media dei voti degli studenti
  - conteggi;
    es: numero di corsi della laurea triennale
  - massimi e minimi

FUNZIONI AGGREGATIVE

- Funzione aggregativa
  - argomento: attributo di una tabella
  - calcolata esaminando i valori di un attributo appartenenti ad ennuple diverse
- Tipicamente
  - SUM (somma), COUNT (conteggio), AVG (media), MIN (minimo), MAX (massimo)

FUNZIONI AGGREGATIVE

- Sintassi
  - si utilizzano nella proiezione
- Esempio:
  - calcolo del voto medio degli esami
  - funzione AVG() applicata all'attributo voto della tabella Esami

FUNZIONI AGGREGATIVE

- Semantica
  - viene calcolato il risultato della proiezione sugli attributi utilizzati come argomenti
  - viene applicata la funzione aggregativa ai valori dell'attributo
  - il risultato è una tabella con una singola ennupla
  - una colonna per ciascuna funzione aggregativa utilizzata nella proiezione

FUNZIONI AGGREGATIVE

- Esempio:

  Risultato = $\pi_{AVG(voto)}$ (Esami)

  - I passo: π_{voto} (Esami)
  - II passo: calcolo della media dei valori dell'attributo voto
  - III passo: viene creata la tabella Risultato con una unica colonna (chiamata AVG(voto)) ed un'unica ennupla, contenente il risultato

FUNZIONI AGGREGATIVE: ESEMPI

- Voto medio degli esami

$$\text{Risultato} = \pi_{\text{AVG(voto)}}(\text{Esami})$$

$$\rho_{\text{AVG(voto) AS votomedio}}(\pi_{\text{AVG(voto)}}(\text{Esami}))$$

Risultato

AVG(voto)
24,7

votomedio
24,7

- Voto massimo e minimo degli esami

$$\rho_{\text{MIN(voto) AS votomin, MAX(voto) AS votomax}}(\pi_{\text{MIN(voto), MAX(voto)}}(\text{Esami}))$$

votomin	votomax
20	30

- Numero di corsi della laurea triennale

$$\rho_{\text{COUNT(cod) AS numcorsi}}(\pi_{\text{COUNT(cod)}}(\sigma_{\text{ciclo='laurea tr.'}}(\text{Corsi})))$$

$$\rho_{\text{COUNT(ciclo) AS numcorsi}}(\pi_{\text{COUNT(ciclo)}}(\sigma_{\text{ciclo='laurea tr.'}}(\text{Corsi})))$$

nel caso del conteggio, l'attributo è indifferente (tranne per valori NULL)

numcorsi
2

FUNZIONI AGGREGATIVE

- Regola
  - in una proiezione possono comparire o solo attributi ordinari, o solo funzioni aggregative
  - altrimenti la semantica non è ben definita
- Esempio
  - "Titolo e numero dei corsi della laurea triennale"

$$\rho_{\text{COUNT(cod) AS numcorsi}}\ (\ \pi_{\text{titolo, COUNT(cod)}}(\ \sigma_{\text{ciclo='laurea tr.'}}(\text{Corsi})))$$

SQL DDL

DDL: TABELLE

- Creazione
  - CREATE TABLE <nome> (<schema>);
- <schema>
  - una o più definizioni di attributo
    - zero o più definizioni di vincoli di tabella
- Definizione di attributo
  - <nomeattributo> <tipo> [<vincoli di colonna>]

VALORI DI DEFAULT

- Nella CREATE TABLE è possibile specificare valori di default per gli attributi

```
CREATE TABLE Studenti (
    matr integer PRIMARY KEY,
cognome text NOT NULL,
nome text NOT NULL,
ciclo text DEFAULT ‘laurea tr.’,
anno integer NOT NULL DEFAULT 1,
relatore char(4)text REFERENCES Professori(cod));
```

VINCOLI DI RIFERIMENTO

- **Vincoli di riferimento di colonna**
  - <attr> <tipo> REFERENCES <chiave est.>
  - es: docente char(4) REFERENCES Professori(cod)

- **Vincoli di riferimento di tabella**
  - FOREIGN KEY (<attributi>)
    REFERENCES <chiave est.>
  - es: se la chiave di Professori è nome, cogn. FOREIGN KEY (provincia, collegio)
    REFERENCES Collegi(provincia, numero)

VINCOLI DI RIFERIMENTO

- **Aggiornamenti in cascata**
  - ON {UPDATE | DELETE} <azione>
- <azione>
  - RESTRICT (è l'azione standard)
  - CASCADE
  - SET NULL
  - SET DEFAULT
  - **es:** docente char(4) REFERENCES Professori(cod)
    ON UPDATE SET NULL;

MODIFICHE ALLO SCHEMA

- Modifiche ad una tabella già esistente
  - ALTER TABLE

- Tre funzioni
  - ridenominazione della tabella
  - aggiunta, ridenominazione ed eliminazione di attributi
  - aggiunta ed eliminazione di vincoli

MODIFICHE ALLO SCHEMA

- **Ridenominazione della tabella**
  - ALTER TABLE <nome> RENAME TO <nuovo nome>;
  - es: ALTER TABLE Professori RENAME TO Docenti;

- **Modifiche agli attributi**
  - ALTER TABLE <nome> RENAME COLUMN <vecchio attributo> TO <nuovo attributo>;
  - ALTER TABLE <nome> ADD COLUMN <nuovo attributo> <tipo>;
  - ALTER TABLE <nome> DROP COLUMN <nome attributo>;

MODIFICHE ALLO SCHEMA

- **Esempio**
  - ALTER TABLE Studenti ADD COLUMN dataNascita DATE;
  - ALTER TABLE Studenti ADD COLUMN luogoNascita VARCHAR(20);
  - ALTER TABLE Studenti ADD COLUMN reddito DECIMAL(8,2);
  - ALTER TABLE Studenti RENAME COLUMN dataNascita TO dataDiNascita;
  - ALTER TABLE Studenti DROP COLUMN dataDiNascita;

MODIFICHE ALLO SCHEMA

- **Semantica dell'aggiunta di colonne**
  - la nuova colonna viene aggiunta allo schema
  - a tutte le ennuple viene aggiunto NULL

- **Semantica dell'eliminazione di colonne**
  - la colonna viene eliminata dallo schema, assieme agli eventuali vincoli relativi

MODIFICHE ALLO SCHEMA

- **Aggiunta di vincoli**
  - ALTER TABLE <nome> ADD CONSTRAINT<nome vincolo> <def. vincolo di tabella>;
  - l'istanza deve rispettare il vincolo

- Esempio
  - **ALTER TABLE Studenti ADD CONSTRAINT cf UNIQUE (nome, cognome,dataNascita, luogoNascita);**

MODIFICHE ALLO SCHEMA

- Eliminazione di vincoli
  - ALTER TABLE <nome> DROP CONSTRAINT <nome vincolo>;
- Esempio
  - **ALTER TABLE Studenti DROP CONSTRAINT cf;**
- Attenzione:
  - molti DBMS non supportano né DROP COLUMN, né DROP CONSTRAINT

DDL: VISTE

- Viste
  - tabelle “virtuali”
  - definite attraverso un’interrogazione
  - possono essere utilizzate come tabelle reali

- Due funzioni fondamentali
  - creazione degli schemi esterni (privatezza dei dati o ristrutturazioni)
  - semplificazione di interrogazioni ricorrenti

DDL: VISTE

- Creazione di viste
  - CREATE VIEW <nome> AS <SELECT>;

- Semantica
  - la vista viene ricalcolata sulla base della sua definizione ogni volta che viene usata

- Eliminazione di viste
  - DROP VIEW <nome>;

DDL: VISTE

- Privatezza: esami senza voti

```
CREATE VIEW EsamiSenzaVoti AS
     SELECT studente, corso
     FROM Esami;
```

```
SELECT * FROM EsamiSenzaVoti;

SELECT *
FROM Studenti, EsamiSenzaVoti
WHERE matr=studente;

DROP VIEW EsamiSenzaVoti;
```

DDL: VISTE

- Compiti ricorrenti: professori e numeri

```
CREATE VIEW ProfessoriNumeri AS
    SELECT codice, nome, cognome, numero
    FROM Professori JOIN Numeri
         ON cod=professore;
```

```
SELECT *
FROM ProfessoriNumeri
ORDER BY cognome, nome;
```

DDL: VISTE

- Differenza tra tabelle e viste
  - le tabelle sono materializzate nella base di dati, le viste no (sono derivate dalle tabelle)
  - schema di una vista = attributi e tipi della select
  - istanza di una vista = risultato della select
  - le tabelle sono aggiornabili, le viste no
  - le viste sono sempre aggiornate e consistenti
  - non hanno impatto sulle prestazioni

ESPRESSIONI

- Possono comparire nella SELECT
- Operandi
  - valori degli attributi
- Operatori (non standard)
  - operatori aritmetici +, -, *, % ed altri
  - funzioni matematiche log, exp, sin, …
  - funzioni su stringhe length, substring, …
  - funzioni su date e tempi

ESPRESSIONI

- Esempio: reddito familiare in lire

```
SELECT cognome, nome, reddito*1936.27
FROM Studenti;
```

- **Esempio: media degli esami dello studente Pasquale Bruno in 110mi**

```
SELECT AVG(voto)/30*110
FROM Studenti JOIN Esami ON cod=studente
WHERE cognome='Bruno' AND
      nome='Pasquale';
```

FINE DELLA LEZIONE