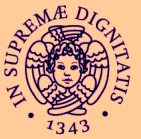
Basi di Dati

SQL-92

Concetti Fondamentali



UNIVERSITÀ DI PISA

Introduzione

- **SQL ("Structured Query Language")**
  - linguaggio per l'interazione con il DBMS
  - tutte le operazioni vengono specificate in SQL
- **DDL ("Data Definition Language")**
  - creazione degli oggetti dello schema
- **DCL ("Data Control Language")**
  - controllo degli utenti e delle autorizzazioni
- **DML ("Data Manipulation Language")**
  - manipolazione dell'istanza della base di dati (interrogazioni e aggiornamenti)

Interrogazioni

- **Istruzione del DML**
  - SELECT
  - sintassi per specificare operatori dell'algebra
- **Filosofia**
  - parzialmente dichiarativa
  - si specificano gli operatori da applicare, non l'ordine in cui devono essere applicati
  - l'ottimizzatore sceglie la strategia ottima

Interrogazioni

- **Forma standard dell'algebra**
  - una o più sottointerrogazioni
  - correlate da operatori insiemistici
- **Sottointerrogazioni**
  - strategia a: prodotti cartesiani tra le tabelle (con eventuali alias)
  - strategia b: join tra le tabelle (con eventuali alias)

Interrogazioni

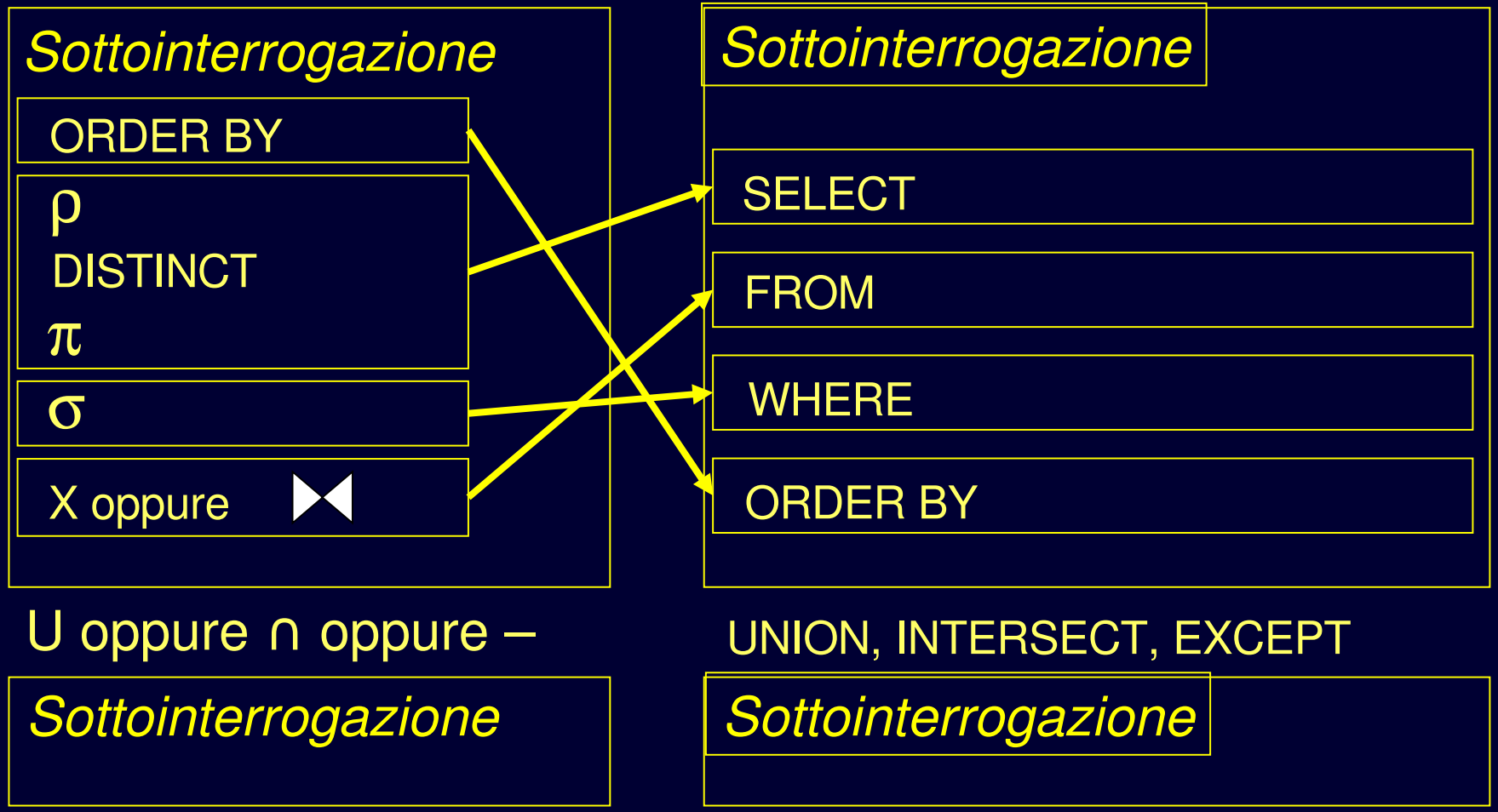
- **Sottointerrogazioni (Continua)**
  - selezioni
  - proiezioni (con funzioni aggregative)
  - eliminazione di duplicati (DISTINCT)
  - ridenominazioni
  - ordinamenti finali (ORDER BY)

Interrogazioni

Interrogazioni

- **Interrogazioni SQL**
  - una o più sottointerrogazioni
  - correlate da operatori insiemistici
- **Sottointerrogazioni: varie "clausole"**
- **Nucleo della SELECT**
  - SELECT: proiezioni, ridenominazioni, distinct
  - FROM: prodotti cartesiani o join, alias
  - [WHERE]: selezioni
- **Clausole aggiuntive**
  - [ORDER BY]: ordinamenti

Interrogazioni



```
TABLE Professori (
   cod char(4) PRIMARY KEY,
   cognome varchar(20) NOT NULL,
   nome varchar(20) NOT NULL,
   qualifica char(15),
   facolta char(10) );

TABLE Studenti (
   matr integer PRIMARY KEY,
   cognome varchar(20) NOT NULL,
   nome varchar(20) NOT NULL,
   ciclo char(20),
   anno integer,
   relatore char(4)
     REFERENCES Professori(cod)
);

TABLE Corsi (
   cod char(3) PRIMARY KEY,
   titolo varchar(20) NOT NULL,
   ciclo char(20),
   docente char(4)
     REFERENCES Professori(cod)
);
```

```
TABLE Tutorato (
   studente integer
     REFERENCES Studenti(matr),
   tutor integer
     REFERENCES Studenti(matr),
     PRIMARY KEY (studente,tutor));

TABLE Esami (
    studente integer
      REFERENCES Studenti(matr)
      ON DELETE cascade
      ON UPDATE cascade,
    corso char(3)
      REFERENCES Corsi(cod),
    voto integer,
    lode bool,
    CHECK (voto>=18 and voto<=30),
    CHECK (not lode or voto=30),
    PRIMARY KEY (studente, corso));

TABLE Numeri (
   professore char(4)
     REFERENCES Professori(cod),
   numero char(9),
  PRIMARY KEY (professore,numero));
```

Esempi

- "Studenti della laurea triennale di anni successivi al primo"

Risultato = $\sigma_{\text{ciclo='laurea tr.' AND anno>1}}(\text{Studenti})$

```
SELECT  *
FROM    Studenti
WHERE   ciclo='laurea tr.' AND anno>1;
ORDER BY
```

Esempi

- "Cognomi e nomi degli studenti"

ElencoNomi = DISTINCT ($\pi_{\text{cognome, nome}}$ (Studenti))

```
SELECT    DISTINCT cognome, nome
FROM      Studenti;
WHERE
ORDER BY
```

Esempi

- **"Cognomi e nomi degli studenti, in ordine alfabetico"**

$$\text{ElencoNomi} = \text{ORDER BY}_{\text{cognome, nome}} (\text{DISTINCT} (\pi_{\text{cognome, nome}} (\text{Studenti})))$$

```
SELECT    DISTINCT cognome, nome
FROM      Studenti
WHERE
ORDER BY  cognome, nome;
```

Esempi

- "Voto medio riportato negli esami"

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

```
SELECT   AVG(voto)
FROM     Esami;
WHERE
ORDER BY
```

Esempi

- **"Cognomi, nomi e numeri di telefono dei professori" (strategia a)**

$$\text{ProfessoriENumeri} = \pi_{\text{cognome, nome, numero}} (\sigma_{\text{cod=professore}} (\text{Professori} \times \text{Numeri}))$$

```
SELECT   cognome, nome, numero
FROM     Professori, Numeri
WHERE    cod=professore;
ORDER BY
```

Esempi

- **“Cognomi, nomi e numeri di telefono dei professori” (strategia b)**

$$\text{ProfessoriENumeri} = \pi_{\text{cognome, nome, numero}} (\text{Professori} \bowtie_{\text{cod=professore}} \text{Numeri})$$

```
SELECT  cognome, nome, numero
FROM    Professori JOIN Numeri
            ON cod=professore;
WHERE
ORDER BY
```

Esempi

- **"Matricola e cognome degli studenti che hanno sostenuto l'esame di informatica teorica" (strategia b)**

$$\text{Risultato} = \pi_{\text{matricola, cognome}} (\sigma_{\text{titolo='Inform. t.'}} (\text{Studenti} \bowtie_{\text{matr=studente}} \text{Esami} \bowtie_{\text{cod=corso}} \text{Corsi}))$$

```
SELECT matricola, cognome
FROM Studenti JOIN Esami ON matr=studente
              JOIN Corsi ON cod=corso
WHERE titolo='Inform. t.';
```

Esempi

- "Cognome e nome delle persone"

$$\text{Risultato} = \rho_{\text{cognome AS cognomePersona, nome AS nomePersona}} (\pi_{\text{cognome, nome}} (\text{Professori})) \cup \pi_{\text{cognome, nome}} (\text{Studenti})$$

```
SELECT cognome AS cognomePersona, nome AS nomePersona
FROM Professori
UNION
SELECT cognome, nome
FROM Studenti;
```

Esempi

- **"Cognome e nome dei professori ordinari che non supervisionano tesi triennali"**

$$\text{Risultato} = \rho_{\text{cognome AS cognomeProf, nome AS nomeProf}} \Big(\pi_{\text{cognome, nome}} \big(\sigma_{\text{qualifica = 'Ordinario'}} (\text{Professori}) \big) \Big) - \pi_{\text{cognome, nome}} \Big(\sigma_{\text{ciclo = 'laurea tr.'}} \big(\text{Studenti} \bowtie_{\text{relatore = cod}} \text{Professori} \big) \Big)$$

Esempi

- “Cognome e nome dei professori ordinari che non supervisionano tesi triennali”

```
SELECT cognome AS cognomeProf, nome AS nomeProf
FROM Professori
WHERE qualifica='ordinario'

EXCEPT

SELECT cognome, nome
FROM Studenti JOIN Professori ON relatore=cod
WHERE ciclo='laurea tr.';
```

Esempi

- **“Cognomi e nomi degli studenti che all’esame di Programmazione hanno riportato un voto superiore a quello dei loro tutor”**
- **Tabelle coinvolte**
  - Studenti, Esami
  - Tutorato, Esami AS EsamiTutor

Esempi

- **“Studenti e tutor” (continua)**

Risultato = $\pi_{\text{cognome, nome}}$ (

$\sigma_{\text{Esami.corso='Pr1' AND EsamiTutor.corso='Pr1' AND Esami.voto > EsamiTutor.voto}}$ (

Studenti $\bowtie_{\text{matr=studente}}$ Esami

$\bowtie_{\text{matr=Tutorato.studente}}$ Tutorato

$\bowtie_{\text{Tutorato.tutor=EsamiTutor.studente}}$ (Esami AS EsamiTutor)))

Esempi

- “Studenti e tutor” (continua)

```
SELECT cognome, nome
FROM Studenti JOIN Esami ON matr=studente
     JOIN Tutorato ON matr=Tutorato.studente
     JOIN Esami AS EsamiTutor
          ON Tutorato.tutor = EsamiTutor.studente
WHERE Esami.corso='Pr1' AND EsamiTutor.corso='Pr1'
          AND Esami.voto > EsamiTutor.voto;
```