


Universita' di Milano Bicocca  
Corso di Basi di dati 1 in eLearning  
C. Batini  
7. SQL DML  
7.1 Select - 1

# Data Manipulation Language (DML): operazioni sui dati

# Avvertenza

- Nel seguito, anche per il linguaggio SQL-DML, saranno descritti gli aspetti piu' rilevanti, lasciando allo studio dello studente gli argomenti di dettaglio, per i quali si rimanda al testo di riferimento, e alcuni esercizi, che non saranno commentati a voce, ma che possono essere seguiti dallo studente per conto proprio. I precedenti argomenti saranno contraddistinti dalla presenza in alto a sinistra del simbolo 

# SQL, due tipi di operazioni sui dati

- Operazione di interrogazione:
  - SELECT
- Operazioni di modifica:
  - INSERT, DELETE, UPDATE

# Operazioni di interrogazione l'operatore `SELECT`

# Due forme per la SELECT

- **SELECT non nidificata**, in cui cioè nella operazione SELECT non è possibile citare una nuova SELECT → è usata in interrogazioni semplici
- **SELECT nidificata**, in cui è possibile nel corpo della SELECT citare una nuova SELECT → è usata in interrogazioni complesse

# L'operazione SELECT non nidificata

# Qualita' di una interrogazione

- Abbiamo visto nella progettazione di schemi concettuali che lo scopo della progettazione e' di costruire uno schema caratterizzato da determinate qualita'.
- Analogamente, nel progettare una interrogazione, dobbiamo sempre avere in mente alcune proprieta', introdotte nella pagina successiva.



# Qualita' di una interrogazione

- **Correttezza** L'interrogazione esprime esattamente la richiesta dell'utente, espresse in linguaggio naturale
- **Efficienza** L'interrogazione viene eseguita in poco tempo e/o con basso utilizzo della memoria.
- **Comprensibilita' (o leggibilita')** L'interrogazione esprime in maniera facilmente comprensibile la richiesta espressa in linguaggio naturale.

# Efficienza di una interrogazione

- In un DBMS colui che progetta la interrogazione non deve preoccuparsi della efficienza. E' infatti il compilatore che utilizzando metodi che verranno studiati in un corso successivo, traduce la `SELECT` in un insieme di comandi che ottimizzano la esecuzione.

# SQL interrogazioni

- Al contrario dell'algebra relazionale, SQL e' un linguaggio dichiarativo: specifica cosa si vuole, non come lo si vuole.
- Come detto, e' il compilatore del DBMS che sceglie la implementazione ottima dal punto di vista dei costi
- Poiche' tuttavia esistono in genere espressioni diverse per la stessa interrogazione in SQL, occorre privilegiare nella scelta la comprensibilita'
- Vedremo piu' avanti degli esempi

# Sintassi e semantica della SELECT

L'istruzione SELECT verrà descritta da due punti di vista:

1. La **Sintassi**, che descrive la forma linguistica che assume la istruzione
2. La **Semantica**, che descrive quali elaborazioni vengono effettuate sui dati quando la SELECT viene eseguita.

# Istruzione SELECT: sintassi

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

- Nella **SELECT**, ListaAttributi e' la "target list", lista obiettivo, cioe' l'insieme degli attributi i cui valori si vuole riprodurre come esito della interrogazione
- La clausola **FROM** contiene le tabelle coinvolte dalla interrogazione
- La clausola **WHERE** contiene una condizione, di forma e significato analoghe a quelle viste nei vincoli di n-pla e nell'algebra relazionale.

# Sintassi della condizione (anche detta formula proposizionale)

- Data una relazione  $r(X)$ , la condizione è una formula ottenuta combinando con i connettivi OR, AND, e NOT condizioni atomiche del tipo
- $A \text{ CONFR } B$  oppure  $A \text{ CONFR } c$ , dove:
  - 1. CONFR è un operatore di confronto ( $=, >, <, \leq, \geq, \neq$ )
  - 2.  $A$  e  $B$  sono attributi in  $X$  sui cui valori CONFR abbia senso (es. intero CONF intero)
  - 3.  $c$  è una costante per cui il confronto CONFR abbia senso (es. COSTO  $\geq$  5)

# Istruzione SELECT: semantica

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

1. Fai il prodotto cartesiano delle tabelle citate nella FROM, poi
2. Seleziona dal prodotto cartesiano le n-ple che soddisfano la condizione presenti nella WHERE, poi
3. Delle n-ple selezionate, fornisci in output i valori degli attributi citati nella ListaAttributi

# Un primo insieme di esercizi introduzione



## Specifiche della base dati di riferimento

- Vogliamo rappresentare un insieme di persone, con nome, età e reddito: assumiamo che il nome sia la chiave primaria (quindi, esiste un solo Franco, una sola Maria, ecc.). Vogliamo rappresentare poi per ogni persona, in due tabelle diverse, il padre e la madre, anche essi con il nome. Anche in questi casi, si assume che il nome identifichi la persona.

## Domanda 7.1.1

- Produci lo schema della base dati relazionale che corrisponde alle precedenti specifiche.

# Risposta

- Tre schemi di relazione:
  - Persone (Nome, Eta', Reddito)
  - Maternita' (NomeMadre, NomeFiglio)
  - Paternita' (NomePadre, NomeFiglio)

## Domanda 7.1.2

- Perché NomeFiglio è chiave nelle due relazioni:
  - Maternità (NomeMadre, NomeFiglio)
  - Paternità (NomePadre, NomeFiglio)

### Risposta

Perché il nome identifica ogni persona, e ogni persona ha una sola madre e un solo padre.

# Concetti introdotti

- Istruzione SELECT non nidificata
- Istruzione SELECT nidificata
- Correttezza della (istruzione) SELECT
- Efficienza della SELECT
- Comprensibilita' (leggibilita') della SELECT
- Sintassi della SELECT
- Semantica della SELECT