Introduzione a SQL

VIDEOCONFERENZA 3b: Panoramica su SQL e prime query

Docente: CHIARA DAMIANI chiara.damiani@unimib.it

SQL

SQL è il linguaggio per la definizione e la manipolazione dei dati in database relazionali, adottato da tutti principali DBMS

originariamente "Structured Query Language", ora "nome proprio

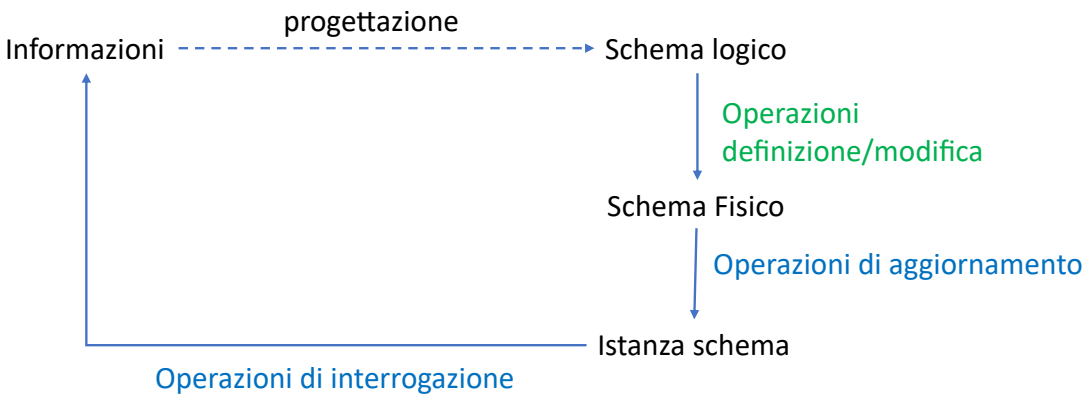
dal 1983 ca. "standard di fatto"

Dal 1986 è uno standard e ne sono state proposte diverse versioni (e.g. SQL-89, SQL-2, SQL-3)

Per la storia di SQL si rimanda al paragrafo 4.1 del libro di riferimento

SQL livelli di conformità

- Nonostante sia uno stantard è stato recepito solo in parte dai vari DBMS
- A seconda delle features che implementano sono stati definiti the livelli di conformità che individuano dei sotto-standard che possono essere realizzati in un dato DBMS
  - ***Entry*** - molto simile a SQL - 89,
  - ***Intermediate*** (versione che soddisfa le esigenze del mercato)
  - ***Full*** (Versione completa anche delle funzioni avanzate che non sono realizzate in alcun DBMS)
- La maggior parte dei database è conforme solo all'Entry Level. Le differenze sono tuttavia di dettaglio.
  - Si veda http://troels.arvin.dk/db/rdbms/ per un confronto



Funzionalità SQL

- L'SQL contiene:
  - **DDL (Data Definition Language)**
    - Operazioni di definizione e modifica schema
  - **DML(Data Manipulation Language)**
    - Operazioni di interrogazione
    - Operazioni di aggiornamento

SQL e Algebra relazionale

- L'SQL è un "relazionalmente completo": ogni espressione dell'algebra relazionale può essere tradotta in SQL
  - …inoltre SQL fa molte altre cose…
- SQL adotta la logica a 3 valori (T,F,Y) dell'Algebra Relazionale
- SQL è un linguaggio computazionalmente completo (e quindi con istruzioni di controllo!) per il supporto di oggetti persistenti…

Algebra relazionale

- Linguaggio prettamente formale che forma la base per linguaggi 'reali'.
- Linguaggio procedurale: si specifica l'algoritmo con cui ottenere il risultato.
- Istruzioni equivalenti possono differire in termini di efficienza.
- Relazioni intese in senso matematico => Insiemi di tuple, definite su attributi
- Negli insiemi non ci possono essere elementi uguali.

Capire l'algebra è la chiave per la comprensione dell'SQL

SQL

- Linguaggio più usato per basi di dati relazionali.
- Linguaggio (parzialmente) dichiarativo: si specifica il risultato da ottenere senza preoccuparsi di specificare l'algoritmo.
- Istruzioni equivalenti differiscono solo per leggibilità.
- Relazioni intese come tabelle.
- Possono esserci righe uguali

Istruzioni SQL => **DBMS** (query optimizer)=> istruzioni ottimizzate per efficienza

Istruzioni principali dell'SQL

Operazioni di definizione schema e modifica

- **CREATE** — Definisce database, tabelle, domini, viste, vincoli, autorizzazioni
- **ALTER** — Modifica attributi e vincoli
- **DROP** — Elimina database e tabelle

Operazioni di interrogazione

- **SELECT** — Formula query come quelle dell'AR o richieste più elaborate

Operazioni di aggiornamento

Possono basarsi sul risultato di una query

- **INSERT** — Inserisce nuove tuple nelle tabelle
- **DELETE** — Elimina tuple nelle tabelle
- **UPDATE** — Modifica tuple

Da preparare Lunedì 20 Aprile con registrazioni del Prof. Batini (Argomento 7)

- Martedì 21 farete esempio pratico in laboratorio

Operazioni di definizione schema e modifica

- **CREATE** — Definisce database, tabelle, domini, viste, vincoli, autorizzazioni
- **ALTER** — Modifica attributi e vincoli
- **DROP** — Elimina database e tabelle

Ci concentriamo inizialmente su operatori AR e poi estendiamo

Operazioni di interrogazione

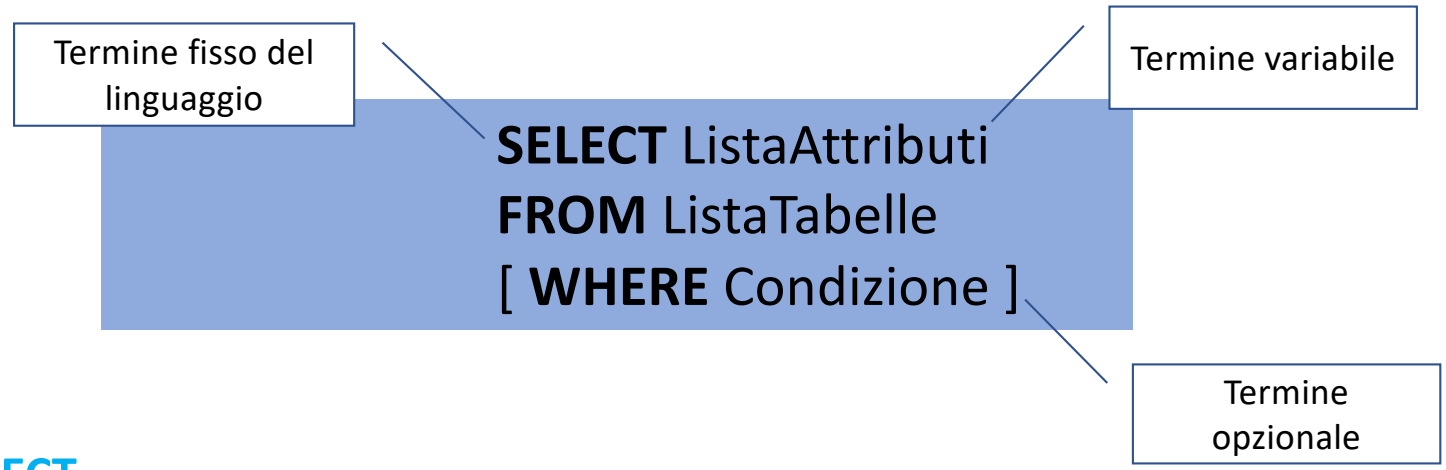
- **SELECT** — Formula query come quelle dell'AR o richieste più elaborate

Operazioni di aggiornamento

Possono basarsi sul risultato di una query

- **INSERT** — Inserisce nuove tuple nelle tabelle
- **DELETE** — Elimina tuple nelle tabelle
- **UPDATE** — Modifica tuple

L’istruzione SELECT: sintassi



- clausola **SELECT** (aka target list) → **PROIEZIONE**, scelgo le colonne
- clausola **FROM** → Tabella/e da cui voglio estrarre le informazioni. Se si indicano più tabelle, viene considerato il loro **prodotto cartesiano,** salvo diversamente specificato
- clausola **WHERE** → **SELEZIONE**, scelgo le righe

Introduzione informale alla sintassi della SELECT

- Impareremo la sintassi facendo degli esempi
- Iniziamo ripetendo gli esercizi fatti per Algebra Relazionale

Dato il seguente schema relazionale:

Personale_non_docente(Matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(Matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Stipendio(Classe,Valore)

ESEMPIO DI PROIEZIONE SENZA SELEZIONE: Selezionare Matricola e Classe Stipendio del Personale Docente

```
SELECT Matricola_d, Classe_stipendio
FROM  Personale_docente
```

Dato il seguente schema relazionale:

Personale_non_docente(Matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(Matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Stipendio(Classe,Valore)

ESEMPIO DI SELEZIONE SENZA PROIEZIONE: Selezionare il personale docente con Classe Stipendio 3

L'asterisco * si usa come abbreviazione per indicare tutti gli attributi

```
SELECT *
FROM  Personale_docente
WHERE Classe Stipendio = 3
```

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esercizio 1: formulare una query SQL che produca Matricola, Cognome e nome dei Ricercatori con classe stipendio 3

QUERY IN AR (Es. 6)

$\Pi_{\text{matricola\_d,Cognome,Nome}}(\sigma_{\textbf{Ruolo='Ricercatore' AND Classe\_stipendio=3}}(\text{Personale\_docente}))$

QUERY IN SQL

?

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esercizio 1: formulare una query SQL che produca Matricola, Cognome e nome dei Ricercatori con classe stipendio 3

QUERY IN AR (Es. 6)

$\Pi_{\text{matricola\_d,Cognome,Nome}}(\sigma_{\text{Ruolo='Ricercatore' AND Classe\_stipendio=3}}(\text{Personale\_docente}))$

QUERY IN SQL

```
SELECT Matricola_d,Cognome,Nome
FROM Personale_docente
WHERE Ruolo='Ricercatore' AND Classe_stipendio=3
```

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esempio di operatori insiemistici: Formulare una query SQL che produca tutti i dipendenti dell'ateneo.

QUERY IN AR (Es. 1)

Personale_non_docente U Personale_docente

QUERY IN SQL

```
SELECT * FROM Personale_non_docente
UNION
SELECT * FROM Personale_docente
```

Algebra relazionale

- Gli operatori insiemistici si applicano solo a relazioni definite sugli stessi attributi
- L'ordine degli attributi è irrilevante

SQL

- Gli operatori insiemistici si applicano a relazioni definite sullo stesso numero di attributi
- L'ordine degli attributi è rilevante

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esercizio 2: Formulare una query che produca tutte le persone dell'ateneo.

QUERY IN AR (Es. 2)

$$\Pi_{matricola\_d,Cognome,Nome}(Personale\_non\_docente) \cup \Pi_{matricola\_d,Cognome,Nome}(Personale\_docente) \cup \rho_{matricola\_d <-- matricola\_st}(\Pi_{matricola\_st,Cognome,Nome}(Studente))$$

QUERY IN SQL

?

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esercizio 2: Formulare una query che produca tutte le persone dell'ateneo.

QUERY IN AR (Es. 2)

$$\Pi_{matricola\_d,Cognome,Nome}(Personale\_non\_docente) \cup \Pi_{matricola\_d,Cognome,Nome}(Personale\_docente) \cup \rho_{matricola\_d <-- matricola\_st}(\Pi_{matricola\_st,Cognome,Nome}(Studente))$$

```
SELECT Matricola_d,Cognome,Nome
FROM Personale_docente
UNION
SELECT Matricola_d,Cognome,Nome
FROM Personale_non_docente
UNION
SELECT Matricola_st,Cognome,Nome
FROM Studente
```

Personale_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
485801	BELOTTI	GIOVANNI	Associato	5
512601	CAMPIGLIA	GIUSEPPE	Ricercatore	6
774002	BUZZI	UMBERTO	Ricercatore	3
94302	QUERCINI	PIETRO	Ordinario	14

Personale_non_docente

Matricola_d	Cognome	Nome	Ruolo	Classe_stipendio
1446203	VISANI	FRANCESCO	Dipendente	C
1246201	BOVONE	LUIGI	Dipendente	C
1157302	FORMAGGIO	PAOLO	Dipendente	B
1280402	COLOMBO	LUCA	Dipendente	C

Studente

Matricola_st	Cognome	Nome	Ruolo
1446203	VISANI	FRANCESCO	F1801Q
1497001	MOSCHINI	PIETRO	F1801Q
1515801	COLOMBO	LUCA	F1801Q
1524501	GATTI	LUIGI	F1801Q

SELECT Matricola_d,Cognome,Nome
FROM Personale_docente

Matricola_d	Cognome	Nome
485801	BELOTTI	GIOVANNI
512601	CAMPIGLIA	GIUSEPPE
774002	BUZZI	UMBERTO
94302	QUERCINI	PIETRO

SELECT Matricola_d,Cognome,Nome
FROM Personale_non_docente

Matricola_d	Cognome	Nome
1446203	VISANI	FRANCESCO
1246201	BOVONE	LUIGI
1157302	FORMAGGIO	PAOLO
1280402	COLOMBO	LUCA

SELECT Matricola_st,Cognome,Nome
FROM Studente

Matricola_st	Cognome	Nome
1446203	VISANI	FRANCESCO
1497001	MOSCHINI	PIETRO
1515801	COLOMBO	LUCA
1524501	GATTI	LUIGI

```
SELECT Matricola_d,Cognome,Nome
FROM universita.Personale_docente
union
SELECT Matricola_d,Cognome,Nome
FROM universita.Personale_non_docente
union
SELECT Matricola_st,Cognome,Nome
FROM universita.studente
```

Matricola_d	Cognome	Nome
485801	BELOTTI	GIOVANNI
512601	CAMPIGLIA	GIUSEPPE
774002	BUZZI	UMBERTO
94302	QUERCINI	PIETRO
1446203	VISANI	FRANCESCO
1246201	BOVONE	LUIGI
1157302	FORMAGGIO	PAOLO
1280402	COLOMBO	LUCA
1497001	MOSCHINI	PIETRO
1515801	COLOMBO	LUCA
1524501	GATTI	LUIGI

Dato il seguente schema relazionale:

Personale_non_docente(matricola_d,Cognome,Nome, Ruolo, Classe_stipendio)
Personale_docente(matricola_d,Cognome,Nome,Ruolo, Classe_stipendio)
Studente(matricola_st,Cognome,Nome, corso_di_Laurea)

Esempio di redonominazione: Formulare una query SQL che produca tutte le persone dell'ateneo.

QUERY IN AR (Es. 2)

$$\rho_{Matr \leftarrow Matricola\_d}(\Pi_{matricola\_d,Cognome,Nome}(Personale\_non\_docente)) \cup \rho_{Matr \leftarrow Matricola\_d}(\Pi_{matricola\_d,Cognome,Nome}(Personale\_docente)) \cup \rho_{Matr \leftarrow Matricola\_st}(\Pi_{matricola\_st,Cognome,Nome}(Studente))$$

QUERY IN SQL

```
SELECT Matricola_d AS Matr ,Cognome,Nome
FROM Personale_docente
UNION
SELECT Matricola_d,Cognome,Nome
FROMPersonale_non_docente
UNION
SELECT Matricola_st,Cognome,Nome
FROM Studente
```

AS operatore di ridenominazione

```
SELECT Matricola_d AS Matr ,Cognome,Nome
FROM universita.Personale_docente
union
SELECT Matricola_d,Cognome,Nome
FROM universita.Personale_non_docente
union
SELECT Matricola_st,Cognome,Nome
FROM universita.studente
```

Matr	Cognome	Nome
485801	BELOTTI	GIOVANNI
512601	CAMPIGLIA	GIUSEPPE
774002	BUZZI	UMBERTO
94302	QUERCINI	PIETRO
1446203	VISANI	FRANCESCO
1246201	BOVONE	LUIGI
1157302	FORMAGGIO	PAOLO
1280402	COLOMBO	LUCA
1497001	MOSCHINI	PIETRO
1515801	COLOMBO	LUCA
1524501	GATTI	LUIGI