

SQL \rightarrow linguaggio DICHIARATIVO

INTERROGAZIONE:

Comando Principale \bar{e} SELECT

~~///~~ SELECT ~~///~~ FROM [TABELLE SORGENTI] WHERE [condizioni]

SELECT

Posso Specificare espressioni che determinano gli Attributi Separati da una Virgola

FROM

dico da Dove faccio la SELECT

WHERE

Condizione BOOLEANA per Selezionare i Valori degli Attributi ma con Alcune Clausole come:

- HAVING
- GROUP BY
- ORDER BY

SELECT mi definisce la Struttura dello Schema Risultante



$\Pi \leadsto$ una proiezione

~~///~~ SELECT [{ EXPRESSION | * }, { ... }] \rightarrow tutti gli Attributi di una Tabella

Posso fare anche un ALIAS (è la RIDEENOMINAZIONE)

[{ EXPRESSION [AS nome_output] | * }]

DISTINCT permette di SELEZIONARE solo le RIGHE con Valori \neq

elimina così i duplicati

→ Visto che SQL non lo fa in Automatico

Dato INSEGNAMENTO (CODICE, NOME, NUM-CREDITI)

CODICE	NOME	NUM-CREDITI
INFØ1	PROG 1	12
INFØ2	DB	12
INFØ3	ANALISI	6

SELECT * FROM Insegnamento; \Rightarrow RESTITUISCE tutto

SELECT Nome FROM Insegnamento \Rightarrow Nome dei CORSI

Codice e nome in Maiuscolo:

SELECT Codice, Upper(Nome) FROM Insegnamento

funzioni Comode:
 Lower
 Upper

CONCATENAZIONE DELLE STRINGHE \Rightarrow [Codice][Nome + CFU]

SELECT codice, nome || ' + CFU ' || num-crediti FROM Insegnamento

CODICE	Nome	Num.	Crediti
INF01	progr 1	CFU	12
:	db	CFU	12
!	Analisi	CFU	6

Ho Questa parte fissa
CFU

Se volessi dare un Nome Significativo
dovrei mettere AS nome-che-voglio

From Nome-Tabella → singola

From Tab₁, ..., Tab_N → multipla

Posso Indicare delle SOTTO-QUERY

Se nel FROM ho più Tabelle viene fatto il prodotto Cartesiano

FROM A, B ←

Avrà dimensione $|A| \cdot |B|$

Ma combinerò Ogni Riga di A con ogni Riga di B

Se ci Sono Attributi con Nome Uguale vengono Visti Come 2
Attributi ≠ e Troveremo [Nome-TAB.Nome-Attributo]

FROM Nome-Tab as Nuova-Nome sarà Solo una Ridenominazione
all' Interno della QUERY

ESEMPIO:

TAB-A(ID, A₁)

TAB-B(ID, B₁)

/// SELECT * FROM TAB-A, TAB-B ed Avrò N Righe dove $N = |A| \cdot |B|$ ed ogni Riga ha 4 Colonne

/// SELECT TAB-A.ID, B₁ FROM TAB-A, TAB-B e così ID non è ambiguo visto che ho Specificato la Tabella dove l'Attributo è in Comune

/// SELECT A.ID, B₁ FROM TAB-B, TAB-A AS A ed ho Ridenominato le Tabelle per Comodità

JOIN:

Scelgo Solo le RIGHE che mi Soddisfano una CONDIZIONE

/// SELECT * FROM A, B WHERE [condition]

/// SELECT * FROM A <JOINTYPE> ON <JOIN-CONDITION>

ESEMPIO:

IMPIEGATO(MATRICOLA, nome, cognome, reparto)

REPARTO(CODICE, nome, Tel)

Impiegato.REPARTO → REPARTO

/// V. Impiegato il Nome, Cognome e Telefono del Reparto:

SELECT Cognome, Impiegato.Nome, Reparto.Tel FROM Impiegato, Reparto
WHERE Impiegato.reparto = Reparto.codice

WHERE è Sempre una Condizione BOOLEANA ricavata combinando condizioni semplici che sono:

- ATTRIB. = COSTANTE
- AND
- ATTR = ATTR
- +
- NOT
- OR

Dove Posso Usare tutti gli OPERATORI DI CONFRONTO
facendo Attenzione che la CLAUSOLA DI ESSERE \neq è $<>$

SELECT Nome FROM Insegnamento WHERE num_cfu \geq 6

CONSIDERANDO ORA: STUD (matricola, nome, cognome, indirizzo, città, media)

MATRICOLA	NOME	COGNOME	INDIRIZZO	CITTÀ	MEDIA
1001	Mario	Rossi	Via Nobile	Verona	27.50
1003	Simone	Mattioli	Via Storto	Rovereto	28.6666

Media Noto che non può Essere né Numeric né decimal

SELECT * FROM stud WHERE cognome = 'Mattioli';

SELECT * FROM stud WHERE cognome = 'Mattioli' AND
città = 'Verona'

NON CASE SENSITIVE

Questa Invece è CASE SENSITIVE

Per Evitare ERRORI/problemi:

LOWER(città) = 'verona'

UPPER(città) = 'VERONA'

oppure

SELECT * FROM stud WHERE NOT cognome = 'Mattioli'

Booleani Sono a 3 Valori in SQL:

- TRUE
- FALSE

Qualsiasi Condizione mi Restituirà NULL - NULL
a Meno che io non INSERISCA:

IS [NOT] NULL

PATTERN MATCHING:

[A] LIKE [PATTERN]



2 caratteri Speciali:

% ⇒ 0...N caratteri Qualsiasi

_ ⇒ 1 carattere Qualsiasi

[A] NOT LIKE [PATTERN]

Città LIKE 'V%R%' ⇒ Città che Iniziano con V ma che Contengono la lettera R

Città LIKE 'V%R_' e Trovo le Città che Iniziano con V e Terminano con R seguito da un carattere Qualsiasi

C'è Anche l'OPERATORE SIMILAR TO

BETWEEN \Rightarrow Testa l'appartenenza ad un **Intervallo**, ad Esempio:

\lll SELECT * FROM Stud WHERE media BETWEEN (21, 27)

\lll SELECT * FROM Stud WHERE media > 21 AND media < 27

Posso Sempre Negare e quindi avere il **NOT BETWEEN**

IN \Rightarrow Permette di Verificare se il Valore di un Attributo è incluso in un Insieme di Valori

\lll SELECT * FROM Stud WHERE nome IN ('Simone', 'Omar', 'David')

ORDINAMENTO:

Una Volta Ottenuto il Risultato, posso Richiedere di averlo rispetto un ORDINAMENTO Ben Specifico tramite la

ORDER BY dove Inserisco 1...N Attributi e dire se

è \lll ASCENDENTE

\lll DECRESCENTE

\downarrow se ho più Attributi allora a Parità di primo eseguo l'ordinamento sul SECONDO

AGGREGAZIONE:

Permettono di determinare un Valore considerando i Valori Ottenuti dalla SELECT con:

- **MIN** \rightarrow

- **COUNT** \rightarrow #righe Risultato

- **MAX** \rightarrow

- AVG →

- SUM →

COUNT(*) ⇒ conta Tutte le Righe del Risultato

SELECT COUNT(*) FROM Stud ⇒ ed è un CONTATORE

COUNT (EXPRESSION)

↓
<nome.attributo>

SELECT COUNT(città) ⇒ # righe in cui Attributo città NON è NULL.

COUNT (DISTINCT <expression>) ⇒ Conta i Valori, SENZA duplicati dove è Vera l'Espressione

MIN/MAX/AVG

SELECT $\begin{matrix} \text{max} \\ \text{min} \\ \text{avg} \\ \text{sum} \end{matrix} \left(\text{media} \right)$ FROM Stud

SUM Somma i Valori Mentre COUNT conta le Righe