**DML**

Insert

L'istruzione INSERT viene utilizzata per inserire nuovi record in una tabella.

INSERT INTO table\_name (*column1, column2, column3, …*) VALUES (*value1, value2, value3, …*);

Update

L'istruzione UPDATE viene utilizzata per modificare i record esistenti in una tabella.

UPDATE table\_name  
SET column1 = value1, column2 = value2, ...  
WHERE condition;

Delete

L'istruzione DELETE viene utilizzata per eliminare i record esistenti in una tabella.

DELETE FROM table\_name WHERE condition;

**QL**

SELECT

Il costrutto select è composto da:

* \_Target list
* \_FROM clause
* \_WHERE clause

Sia nella target list che nella from possono essere inseriti gli alias.

SELECT column1  
FROM table1 AS t1  
WHERE condizione;

Operatori BETWEEN e LIKE

L’operatore BETWEEN serve per verificare che un attributo sia in un certo intervallo.

SELECT column1  
FROM table1 AS t1  
WHERE t1.valore BETWEEN 160 AND 170;

L’operatore LIKE serve per effettuare una ricerca su stringhe o caratteri simili.

SELECT column1, column2  
FROM table1 AS t1  
WHERE t1.column1 LIKE R\* OR t1.column2 LIKE G\*;

Operatori IN e IS NULL

L’operatore IN confronta un attributo con i valori contenuti all’interno dell’operando.

SELECT column1  
FROM table1 AS t1  
WHERE t1.column1 IN (*1, 2, 3*);

L’operatore IS NULL serve per vedere se ho un’istanza con un attributo nullo.

SELECT  column1  
FROM table1 AS t1  
WHERE t1.column1 IS NULL;

Join

Il costrutto join serve a combinare i dati di due o più tabelle basandosi su alcune condizioni logiche.

Esistono diversi tipi di JOIN:

* INNER JOIN prende solo i record in comune.
* LEFT JOIN prende anche i record non comuni della prima tabella nominata.
* RIGHT JOIN prende anche i record non comuni della seconda tabella nominata.
* FULL JOIN prende tutti i record non comuni alle tabelle.

SELECT column1, column2  
FROM table1 AS t1 INNER JOIN table2 AS t2 ON t1.column1 = t2.column1 WHERE condition;

SELECT column1, column2  
FROM table1 AS t1, table2 AS t2 WHERE t1.column1 = t2.column1 AND condition;

Funzioni di aggregazione

L’operatore COUNT serve per contare le istanze che ha un’entità (conta anche i valori null).

SELECT COUNT (*column1*) AS tot.  
FROM table\_name;

L’operatore AVG serve per calcolare la media.

SELECT AVG (*column1*) AS average.  
FROM table\_name;

L’operatore MIN restituisce il valore più basso contenuto nella colonna in questione.

SELECT MIN (*column1*) AS min  
FROM table\_name;

L’operatore MAX restituisce il valore più alto contenuto nella colonna in questione.

SELECT MAX (*column1*) AS max  
FROM table\_name;

L’operatore SUM restituisce la somma dei valori contenuti in una colonna.

SELECT SUM (*column1*) AS sum  
FROM table\_name;

Order by

La parola chiave ORDER BY viene utilizzata per ordinare il set di risultati in ordine crescente o decrescente.

Per ordinare i record in ordine decrescente, utilizzare la parola chiave DESC.

SELECT column1, column2, ...  
FROM table\_name  
ORDER BY column1, column2, ... ASC|DESC;

Group by e having

L'istruzione GROUP BY raggruppa righe con gli stessi valori in righe di riepilogo, l'istruzione GROUP BY viene spesso utilizzata con funzioni aggregate (COUNT, MAX, MIN, SUM, AVG) per raggruppare il set di risultati per una o più colonne.

SELECT column\_name(s)  
FROM table\_name  
WHERE condition  
GROUP BY column\_name(s);

La clausola HAVING è stata aggiunta a SQL perché non è stato possibile utilizzare la parola chiave WHERE con le funzioni di aggregazione. Si usa solo quando è presente una group by.

SELECT column\_name(s)  
FROM table\_name  
WHERE condition  
GROUP BY column\_name(s)  
HAVING condition  
ORDER BY column\_name(s);

Le subquery

Le subquery o query annidate sono semplicemente delle query dentro altre query.

SELECT column\_name(s)  
FROM table\_name  
WHERE condition > ( SELECT column\_name(s)  
 FROM table\_name  
 WHERE condition );