**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\_nameSET column1 = value1, column2 = value2, ...WHERE condition;

**Delete**

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

DELETE FROM table\_nameWHERE condition;

**QL**

**Select**

Il costrutto select è composto da:

* \_Target list SELECT attributo1, attributo2
* \_FROM clause FROM tabella1 t1
* \_WHERE clause WHERE condizione;

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

**Operatori between e like**

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

SELECT attributo1, attributo2

FROM tabella1 AS t1

WHERE t1.valore BETWEEN 160 AND 170;

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

SELECT attributo1, attributo2

FROM tabella1 AS t1

WHERE t1.parola1 LIKE R\* OR t1.parola2 LIKE G\*;

**Operatori in e is null**

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

SELECT attributo1, attributo2

FROM tabella1 AS t1

WHERE t1.valore IN (1, 2, 3);

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

SELECT attributo1, attributo2

FROM tabella1 AS t1

WHERE t1.valore 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.

**Sintassi**:

SELECT attributo1, attributo2

FROM tabella1 AS t1, tabella2 AS t2

WHERE t1.attributo1 = t2.attributo1 AND condizioni;

SELECT attributo1, attributo2

FROM tabella1 AS t1 INNER JOIN tabella2 AS t2 ON t1.attributo1 = t2.attributo1

WHERE condizioni;

**Funzioni di aggregazione**

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

SELECT COUNT(attributo1) AS tot

FROM tabella1;

L’operatore AVG serve per calcolare la media.

SELECT AVG(attributo1) AS media

FROM tabella1;

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

SELECT MIN(attributo1) AS minimo

FROM tabella1;

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

SELECT MAX(attributo1) AS massimo

FROM tabella1;

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

SELECT SUM(attributo1) AS somma

FROM tabella1;

**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\_nameWHERE 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 conditionGROUP BY column\_name(s)HAVING conditionORDER BY column\_name(s);

**Le subquery**

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

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