SQL Data Manipolation Language

Istruzioni di base

DML

La manipolazione dei dati in SQL avviene principalmente tramite 4 parole chiave:

- SELECT: Permette di recuperare i dati dal db eseguendo interrogazioni (query).Ha numerosi parametri che permettono di eseguire query anche molto complesse. Per questo motivo si parla anche di DQL (Data Query Language).
- INSERT: Permette di inserire nuovi dati nel database. Può anche utilizzare il risultato di una query di SELECT.
- DELETE: Permette di cancellare dati esistenti.
- UPDATE: Permette di modificare uno o più campi dei dati esistenti

DELETE e UPDATE fanno uso di condizioni avanzate per determinare quali dati cancellare/modificare. Sono le stesse condizioni utilizzate da SELECT

DML – Selezione dei record

La selezione di record da una o più tabelle (interrogazione / query) avviene tramite il costrutto SELECT.

SELECT ha numerosi parametri che permettono la realizzazione di interrogazioni anche molto complesse. Nella sua forma più semplice è:

```
SELECT <elenco di campi>

FROM <elenco di tabelle>

WHERE <condizioni>
```

L'istruzione lavora sulle tabelle specificate nella clausola FROM e sulle sole righe che soddisfano la condizione impostata nella clausola WHERE. Vengono restituiti solamente i campi specificati nella clausola SELECT.

La clausola WHERE non è obbligatoria. In tal caso vengono restituite tutte le righe.

DML – Selezione dei record

Esempio:

```
SELECT Name, Sex
FROM Athletes
WHERE NOC = 'ITA'
```

Se vogliamo tutti i campi possiamo evitare di scriverli tutti, utilizzando l'operatore *

```
SELECT *
FROM Athletes
WHERE NOC = 'ITA'
```

L'operatore * restituisce i campi nello stesso ordine con cui sono stati definiti nella tabella.

DML – Selezione dei record senza duplicati

Come detto in precedenza, le tabelle e le query SQL permettono righe duplicate. Cosa non possibile con le relazioni.

Per fare in modo che il risultato di una query non comprenda i valori duplicati utilizziamo la parola chiave DISTINCT:

```
SELECT DISTINCT Name, Sex
FROM Athletes
WHERE NOC = 'ITA'
```

DML – Espressioni in SELECT

L'elenco dei campi inserito nella clausola SELECT può anche contenere espressioni, utili per restituire valori differenti rispetto a quanto memorizzato nel database.

```
SELECT DISTINCT Name, Sex, Height*0.0328084, Weight*2.20462
FROM Athletes
WHERE NOC = 'ITA'
```

I campi calcolati tramite espressioni non hanno un nome significativo. Possiamo assegnarlo manualmente.

DML – Rinominare le colonne

E’ possibile associare un nome arbitrario alle colonne inserite nella clausola SELECT. Utile per dare nomi più significativi e/o per dare un nome alle colonne calcolate.

Ciò è possibile tramite l’operatore AS posto dopo la colonna a cui si riferisce.

```
SELECT DISTINCT Name, Sex, Height*0.0328084 AS HeightFt,
Weight*2.20462 AS WeightLb

FROM Athletes

WHERE NOC = 'ITA'
```

DML – Clausola WHERE

Nella clausola WHERE possiamo costruire espressioni complesse per determinare quali righe della tabella devono essere comprese nel risultato della query.

Gli operatori di base sono quelli propri dell'algebra booleana: =, <>, <, <=, >, >=, AND, OR, NOT

E' possibile anche l'utilizzo delle parentesi tonde e degli operatori algebrici + - * /

```
SELECT *
FROM Athletes
WHERE NOC = 'ITA' AND Year = 2016
```

```
SELECT *
FROM Athletes
WHERE NOC = 'ITA' AND Year >= 1980 AND Year < 2000
```

DML – Clausola WHERE

Esistono anche altri operatori per migliorare l'espressività delle query. Vedremo:

- BETWEEN
- IN
- LIKE
- IS / IS NOT NULL

DML – Clausola WHERE

BETWEEN permette di specificare un range di valori per restituire tutti le righe che ne fanno parte (o che non ne fanno parte).

BETWEEN ha lo stesso effetto di >= … AND <= …

```
SELECT *
FROM Athletes
WHERE NOC = 'ITA' AND Year BETWEEN 1980 AND 1999
```

Per ottenere le righe non comprese nell'intervallo si utilizza NOT BETWEEN

L'operatore funziona anche con stringhe. In tal caso l'ordinamento è quello alfabetico.

DML – Clausola WHERE

IN permette di specificare un elenco di valori per restituire tutti le righe che ne fanno parte (o che non ne fanno parte).

IN ha lo stesso effetto di = … OR = … OR = …

L'elenco di valori va espresso tra parentesi tonde, separati da virgola.

```
SELECT *

FROM Athletes

WHERE NOC IN ('ITA','USA')
```

Per ottenere le righe non comprese nell'elenco si utilizza NOT IN

DML – Clausola WHERE

LIKE permette di effettuare ricerche su parti di un campo testuale. Tramite caratteri speciali è possibile determinare dei "pattern" di ricerca sui campi:

- ? Un singolo carattere
- % Una stringa di 0 o più caratteri

```
SELECT *
FROM Athletes
WHERE Name LIKE '%Phelps%'
```

Per ottenere le righe non soddisfano il pattern di ricerca si utilizza NOT LIKE

DML – Clausola WHERE

Tutti gli operatori visti in precedenza IGNORANO i campi con valore NULL.

Per considerarli, o per escluderli, utilizziamo rispettivamente gli operatori IS NULL e IS NOT NULL

```
SELECT *
FROM Athletes
WHERE Medal IS NOT NULL
```

Esercizio: Query di Selezione / 1

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

1. Selezione di tutti gli atleti italiani che hanno gareggiato nel nuoto (Swimming) nelle olimpiadi del 1952
2. Dettagliare meglio la query precedente per restituire solo gli atleti di sesso maschile che hanno gareggiato nel 100m style libero (usare like)
3. Scoprire se i giochi del 1998 erano estivi o invernali. La query deve restituire 1 riga e 1 colonna.
4. Selezionare tutti gli atleti che hanno vinto una medaglia ad una età compresa tra 13 e 15 anni
5. Recuperare tutte le edizioni dei giochi svolte a Los Angeles o a London

Esercizio: Query di Selezione / 1 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

1. Selezione di tutti gli atleti italiani che hanno gareggiato nel nuoto (Swimming) nelle olimpiadi del 1952

```
USE Olympics;

SELECT *

FROM Athletes

WHERE NOC = 'ITA' AND Sport = 'Swimming' AND Year = 1952
```

Esercizio: Query di Selezione / 1 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

2. Dettagliare meglio la query precedente per restituire solo gli atleti di sesso maschile che hanno gareggiato nel 100m style libero (usare like)

```
USE Olympics;

SELECT *

FROM Athletes

WHERE NOC = 'ITA' AND Sport = 'Swimming' AND Year = 1952 AND Sex='M' AND Event LIKE
'%100%metres%free%'
```

Esercizio: Query di Selezione / 1 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

3. Scoprire se i giochi del 1998 erano estivi o invernali. La query deve restituire 1 riga e 1 colonna.

```
USE Olympics;

SELECT DISTINCT Season

FROM Athletes

WHERE Year = 1998
```

Esercizio: Query di Selezione / 1 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

4. Selezionare tutti gli atleti che hanno vinto una medaglia ad una età compresa tra 13 e 15 anni

```
USE Olympics;

SELECT DISTINCT Name

FROM Athletes

WHERE Age BETWEEN 13 AND 15 AND Medal IS NOT NULL
```

Esercizio: Query di Selezione / 1 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

5. Recuperare tutte le edizioni dei giochi svolte a Los Angeles o a London

```
USE Olympics;

SELECT DISTINCT Games,City

FROM Athletes

WHERE City IN ('London','Los Angeles')
```

DML – Ordinare i risultati

L'ordine con cui i risultati vengono restituiti non è fisso. Dipende da come sono organizzati i dati fisici sul database.

Per imporre un determinato ordinamento si utilizza la clausola ORDER BY seguita da un elenco di campi su sui ordinare.

L'ordinamento avverrà sul primo campo inserito e, a parità di questo sul secondo, e così via.

L'ordinamento di default e crescente (ASC). Per imporre l'ordinamento decrescente si utilizza la parola chiave DESC

```
SELECT *
FROM Athletes
WHERE Medal IS NOT NULL
ORDER BY Year, Age DESC
```

DML – Ordinare i risultati

I parametri inseriti in ORDER BY possono essere elencati per nome (come già visto) o per "posizione" nell'elenco di selezione. Questa modalità è possibile solamente se i parametri sono specificati in maniera esplicita e non si utilizza SELECT *

```
SELECT Id,Nome,Year,Age
FROM Athletes
WHERE Medal IS NOT NULL
ORDER BY 3, Age DESC
```

La clausola ORDER BY così scritta è equivalente a:

```
ORDER BY Year, Age DESC
```

DML – Limitare il numero dei risultati

Talvolta è comodo limitare il numero di risultati restituiti rispetto a quando previsto dalla clausola WHERE. Può essere utilizzato, per esempio, a fini di debug, per verificare che la query che stiamo scrivendo sia corretta.

Tramite la clausola TOP(n), inserita nella clausola di SELECT, è possibile specificare il numero massimo di righe da restituire.

```
SELECT TOP(10) *
FROM Athletes
WHERE Medal IS NOT NULL
```

DML – Limitare il numero dei risultati

E' possibile limitare il numero di risultati dopo averne "scartati alcuni". E' utile per realizzare versioni paginate (soprattutto quando le query sono eseguite da codice).

```
SELECT *

FROM Athletes

WHERE Medal IS NOT NULL

ORDER BY Id

OFFSET 10 ROWS

FETCH NEXT 10 ROWS ONLY
```

La query restituisce 10 righe a partire dalla 11°. NB: Order by è obbligatorio e non può essere combinata con TOP

Esercizio: Query di Selezione / 2

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

1. Selezione i 10 atleti più giovani ad aver vinto una medaglia d'oro
2. Selezione i 10 atleti più anziani ad aver vinto una medaglia d'oro
3. Selezionare lo sport e l'evento a cui ha partecipato l'atleta più anziano, nei giochi dal 2000 in poi
4. Recuperare l'elenco delle città che hanno ospitato i giochi, in ordine alfabetico
5. Ottenere l'elenco dei partecipanti (solo il nome), in ordine alfabetico e suddivisi per anno e nazione

Esercizio: Query di Selezione / 2 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

1. Selezione i 10 atleti più giovani ad aver vinto una medaglia d'oro

```
USE Olympics;

SELECT DISTINCT TOP(10) Name, Age, NOC, Sport

FROM Athletes

WHERE Medal = 'Gold' AND Age IS NOT NULL

ORDER BY Age
```

Esercizio: Query di Selezione / 2 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

2. Selezione i 10 atleti più anziani ad aver vinto una medaglia d'oro

```
USE Olympics;

SELECT TOP(10) DISTINCT Name, Age, NOC, Sport

FROM Athletes

WHERE Medal = 'Gold' AND Age IS NOT NULL

ORDER BY Age DESC
```

Esercizio: Query di Selezione / 2 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

3. Selezionare lo sport e l'evento a cui ha partecipato l'atleta più anziano, nei giochi dal 2000 in poi

```
USE Olympics;

SELECT  TOP(1) Sport, Event

FROM Athletes

WHERE Year>=2000 AND Age IS NOT NULL

ORDER BY Age DESC, Name
```

Esercizio: Query di Selezione / 2 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

4. Recuperare l'elenco delle città che hanno ospitato i giochi, in ordine alfabetico

```
USE Olympics;

SELECT DISTINCT City

FROM Athletes

ORDER BY City
```

Esercizio: Query di Selezione / 2 - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, scrivere le seguenti query di selezione:

5. Ottenere l'elenco dei partecipanti (solo il nome), in ordine alfabetico e suddivisi per anno e nazione

```
USE Olympics;

SELECT DISTINCT Name

FROM Athletes

ORDER BY Year, NOC, Name
```

DML – Inserimento di righe

L'inserimento di righe all'interno di una tabella esistente avviene tramite il comando INSERT INTO, specificando l'elenco dei valori da inserire in ogni campo della nuova riga.

```
INSERT INTO Nations(Id, Name, NOC) VALUES(1,'Italy','ITA')
```

La lista degli attributi può essere omessa. In tal caso viene considerato l'ordine di creazione.

```
INSERT INTO Nations VALUES(1,'Italy','ITA')
```

Gli attributi non compresi nella lista assumono il valore di default (se previsto), oppure il valore NULL (se ammesso).

```
INSERT INTO Nations(Name, NOC) VALUES('Italy','ITA')
```

DML – Inserimento di righe

L'inserimento contemporaneo di più righe avviene combinando INSERT INTO con SELECT, ovvero inserendo nella tabella il risultato di un'altra query.

```
INSERT INTO Nations(NOC)

SELECT DISTINCT NOC

FROM Athletes
```

I nomi dei campi della query di SELECT possono anche essere differenti rispetto ai nomi dei campi della tabella di destinazione. L'importante è che abbiano un tipo di dato compatibile: il primo campo della query di select deve essere compatibile con il primo campo della query di insert…

DML – Inserimento di righe

INSERT INTO pretende che la tabella in cui inserire le nuove righe sia esistente. Se non esiste va precedentemente creata con CREATE TABLE.

Per effettuare le due operazioni contemporaneamente si utilizza SELECT INTO

```
SELECT DISTINCT NOC
INTO Nations
FROM Athletes
```

La nuova tabella viene creata deducendo nomi e tipi dalla query di select. Non vengono creati vincoli di nessun tipo, che pertanto vanno integrati manualmente.

SELECT INTO genera errore se la tabella esiste.

DML – Cancellazione di righe

La cancellazione avviene tramite l'istruzione DELETE nella quale si specifica la tabella su cui operare e una clausola WHERE tramite la quale selezionare le righe da cancellare.

```
DELETE FROM Nations

WHERE NOC IS NULL
```

La clausola WHERE è identica a quanto giù visto per SELECT.

La cancellazione di una riga potrebbe portare alla violazione di qualche vincolo di integrità referenziale. Vedremo più avanti come gestire questi casi.

DML – Aggiornamento di righe

L'aggiornamento avviene tramite l'istruzione UPDATE nella quale si specifica la tabella su cui operare, una clausola WHERE tramite la quale selezionare le righe da aggiornare, e i valori aggiornati da assegnare a uno o più campi.

```
UPDATE Athletes

SET Medal = NULL

WHERE Medal NOT IN ('Gold','Silver','Bronze')
```

La clausola WHERE è identica a quanto già visto per SELECT.

Anche l'aggiornamento di una riga potrebbe portare alla violazione di qualche vincolo di integrità referenziale.

Esercizio: Query di Inserimento

Con riferimento alla tabella Athletes del db Olympics, e alle tabelle normalizzate precedentemente create:

1. Scrivere le query di inserimento che, leggendo i dati dalla tabella Athletes, riempie le tabelle normalizzate.

NB: Per il momento lavorare solamente sulle tabelle normalizzate che non contengono chiavi esterne. Le altre le affronteremo più avanti.

Esercizio: Query di Inserimento - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, e alle tabelle normalizzate precedentemente create:

1. Scrivere le query di inserimento che, leggendo i dati dalla tabella Athletes, riempie le tabelle normalizzate.

```
USE Olympics;

DELETE FROM Events;

INSERT INTO Events(Event,Sport)

SELECT DISTINCT Event,Sport FROM Athletes;
```

Esercizio: Query di Inserimento - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, e alle tabelle normalizzate precedentemente create:

1. Scrivere le query di inserimento che, leggendo i dati dalla tabella Athletes, riempie le tabelle normalizzate.

```
USE Olympics;

DELETE FROM Games;

INSERT INTO Games(Games, Year, Season)

SELECT DISTINCT Games, Year, Season FROM Athletes;
```

Esercizio: Query di Inserimento - Soluzioni

Con riferimento alla tabella Athletes del db Olympics, e alle tabelle normalizzate precedentemente create:

1. Scrivere le query di inserimento che, leggendo i dati dalla tabella Athletes, riempie le tabelle normalizzate.

```
USE Olympics;

DELETE FROM AthletesNF;

INSERT INTO AthletesNF(IdAthlete, Name, Sex, Height, Weight)

SELECT DISTINCT IdAthlete, Name, Sex, Height, Weight FROM Athletes;
```