Linguaggio SQL

Il linguaggio SQL consente di:

- definire la struttura delle tabelle di un Data base (DDL)
- modificare i dati nelle tabelle, consentendo inserimenti, modifiche, cancellazioni di record (DML)
- gestire il controllo degli accessi, organizzare i dati su memoria di massa, creare gli indici (DMCL)
- interrogare il Data Base

Gestione delle tabelle

Creazione della struttura

```
CREATE TABLE nome_tabella
        ( nome_campo  tipo_campo,
                “                “        ,
                “                “        ,
                “                “        ,
                “                “        );
```

Modifica della struttura

Aggiunta di un campo

```
ALTER TABLE nome_tabella
        ADD nome_campo  tipo_campo;
```

Eliminazione di un campo

```
ALTER TABLE nome_tabella
        DROP nome_campo ;
```

Creazione di un indice

```
CREATE INDEX nome_indice
        ON nome_tabella (campo1,campo2,….);
CREATE UNIQUE INDEX nome_indice
        ON nome_tabella (campo1,campo2,….);
```

Eliminazione di una tabella o di un indice

```
DROP nome_tabella;
DROP nome_indice ON nome_tabella;
```

Manipolazione dei dati

Inserimento di un nuovo record nella tabella

```
INSERT INTO nome_tabella
        (campo1,campo2,campo3,……)
VALUES
        (valore_campo1,valore_campo2,valore_campo3,……):
```

Modifica di uno o più campi

```
UPDATE nome_tabella
        SET nome_campo=valore_campo
        WHERE campo_controllo=valore_controllo;
```

Es. Aggiornare al valore 6 il livello del dipendente con matricola AA345 nella tabella Personale

```
UPDATE  Personale
        SET livello=6
        WHERE matricola=’AA345’;
```

Eliminazione di un record

```
DELETE FROM nome_tabella
        WHERE campo_controllo=valore_controllo ;
```

Modifiche con calcolo

Es. aumentare lo stipendio se il livello è >5

```
UPDATE Personale
        SET stipendio=stipendio*1.05
        WHERE livello>5 ;
```

Eliminazione condizionata di uno o più record

Es. Cancellare tutti i record da Personale se stipendio <750

```
DELETE FROM Personale
        WHERE stipendio<750 ;
```

Comando SELECT

Struttura generale

```
SELECT campo1,campo2,……
        FROM tabella1, tabella2, tabella3,……
        WHERE condizioni di filtro ;
```

Es. Selezionare tutti gli impiegati e visualizzare cognome, nome, codice_fiscale

```
SELECT cognome, nome, codice_fiscale
        FROM Personale
        WHERE funzione=’Impiegato’ ;
```

Se si vogliono tutti i campi

```
SELECT *
        FROM Personale
        WHERE funzione= ‘Impiegato’ ;
```

Per elencare i record evitando i record ripetuti

```
SELECT DISTINCT campo1, campo2, ……
        FROM nome_tabella ;
```

Es. Elenco delle funzioni del personale

```
SELECT DISTINCT funzione
        FROM Personale ;
```

Se nella clausola WHERE si vuole inserire una variabile

```
SELECT Cognome, Nome, Codice_Fiscale
        FROM Personale
        WHERE funzione=[tipoFunzione] ;
```

quando viene eseguita la Query, Access chiede in una finestra di dialogo di inserire una stringa che sarà usata per filtrare i record;

Il comando SELECT usato per eseguire le operazioni relazionali

Selezione

Es. Selezione di Personale per funzione=’Dirigente’

```
SELECT *
        FROM Personale
        WHERE funzione=’Dirigente’ ;
```

Proiezione

Es. Proiezionje di Personale su cognome, nome, codice_fiscale

```
SELECT Cognome, Nome, Codice_fiscale
        FROM Personale ;
```

Congiunzione

Es. Congiunzione di Personale su Filiale e di TabFiliali su Cod_Filiale

```
SELECT *
        FROM Personale, TabFiliali
        WHERE Personale.Filiale= TabFiliali.Cod_Filiale ;
```

Es. Elenco dei dipendenti che hanno funzione di impiegato con cognome, nome, descrizione e indirizzo della filiale dove lavorano

```
SELECT Cognome, Nome, Descrizione, Indirizzo
        FROM Personale, TabFiliali
        WHERE Filiale=Cod_Filiale
        AND funzione=’Impiegato’ ;
```

Usando la forma descrittiva per evidenziare le tre operazioni utilizzate:

```
PROIEZIONE DI
        (CONGIUNZIONE DI
                (SELEZIONE DI Personale su funzione=’Impiegato’)
        su filiale e di TabFiliali su Cod_fliliale)
su Cognome, Nome, Desrizione, Indirizzo ;
```

Selezione con output in una nuova tabella

```
SELECT * INTO nuova_tabella
        FROM  nome_tabella
        WHERE nome_campo=valore ;
```

Es: elenco dei dirigenti in Manager

```
SELECT * INTO Manager
        FROM Personale
        WHERE funzione=’Dirigente’ ;
```

Cardinalità di una relazione

```
SELECT COUNT(*)
        FROM nome_tabella ;
```

Selezionando il nome di un campo si otterrà il numero dei record per i quali il valore del campo non è NULL

```
SELECT COUNT (livello)
        FROM Personale ;
```

darà il numero dei dipendenti ai quali è stato assegnato un livello.

Es. : si vuole conoscere il numero dei dipendenti che lavorano nella provincia di Milano

```
SELECT COUNT(*)
        FROM Personale
        WHERE provincia=’MI’ ;
```

Ordinamenti

SELECT campo1,campo2,……..
FROM nome_tabella
ORDER BY campo1,campo2,……. ;

per default viene effettuato un ordinamento in ordine crescente per ogni campo indicato (ASC). Nel caso in cui si desideri un ordinamento descescente per un determinato campo, occorrerà far seguire il nome del campo dalla parola DESC.

Es. Elencare cognome, nome e data_nascita ordinando in ordine crescente di cognome e descrescente di data_nascita:

```
SELECT Cognome, Nome, data_nascita
    FROM Personale
    ORDER BY Cognome ASC, data_nascita DESC ;
```

Raggruppamenti

Si usano per raggruppare un insieme di righe aventi lo stesso valore nella colonna specificata e ottenere una sola riga per ogni raggruppamento.

Es.: Elencare i livelli esistenti tra i dipendenti con funzione impiegato con N° di dipendenti per ogni livello:

```
SELECT livello, COUNT(livello) as conteggio
    FROM Personale
    WHERE funzione='Impiegato'
    GROUP BY livello ;
```

Es. : Elenco delle funzioni dei dipendenti con lo stipendio medio per ciascuna funzione, dopo aver raggruppato i dipendenti per funzione:

```
SELECT funzione, AVG(Stipendio_base) as Stipendio_medio
    FROM Personale
    GROUP BY funzione ;
```

Es. : Stesso esempio precedente con la condizione che per ogni funzione ci siano più di 2 impiegati:

```
SELECT funzione, AVG(Stipendio_base) as stripendio_medio
    FROM Personale
    GROUP BY funzione
    HAVING COUNT(*) > 2 ;
```

Condizioni di ricerca.

BETWEEN

Operatore che controlla se un valore è compreso tra due valori, compresi gli estremi:

Es.: elenco dei dipendenti con data_assunzione compresa tra 01/01/1995 e 31/12/1999:

```
SELECT Cognome, nome, funzione
    FROM  Personale
    WHERE data_assunzione BETWEEN #01/01/1995# AND #31/12/1999# ;
```

IN

Operatore che indica se un valore appartiene a un insieme di valori specificato.

Es. : Elenco dei dipendenti che risiedono nelle provincie di Bari, Taranto, Lecce:

```
SELECT *
    FROM Personale
    WHERE Provincia IN ('TA','BR','LE') ;
```

LIKE

Confronta il valore di un attributo di tipo carattere con un modello di stringa, che può contenere più caratteri jolly (o metacaratteri).
I caratteri jolly sono :

- _ (underscore) per indicare un carattere qualunque in quella posizione nella stringa;
- % (percentuale) per indicare una sequenza di caratteri in quella posizione della stringa;

Es:

LIKE ‘ALB%’ vengono elencate tutte le stringhe che iniziano con ‘ALB’
LIKE ‘ %RO’ vengono ricercate tutte le stringhe che finiscono con ‘RO’
LIKE ‘%AS%’ vengono ricercate tutte le stringhe che contengono al loro interno I caratteri ‘AS’.
LIKE ‘_ASA’ vengono ricercate tutte le stringhe di 4 caratteri che finiscono con ‘ASA’.
Se non vengono usati caratteri jolly LIKE equivale a = (uguale).

Es. : Elenco cognome nome dei dipendenti il cui cognome inizia con ROS

```
SELECT Cognome, Nome
        FORM Personale
        WHERE Cognome LIKE ‘ROS%’ ;
```