crea una tabella

```
CREATE TABLE Studente(
        ID int,
        Cognome varchar(255),
        Nome varchar(255),
        DataDiNascita date,
        AltezzaInMetri decimal(3,2),
        PesoInChilogrammi decimal(6,3)
)
```

inserisce nella tabella i dati

```
INSERT INTO Studente(ID,Cognome,DataDiNascita,AltezzaInMetri,Telefono)(nomi colonne)

VALUES (0, 'Navarette', '2005-04-12', 1.78, '021064423')(valori inseriti)
```

serve per visualizzare la tabella

```
SELECT *FROM Studente
```

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

```
UPDATE Studente
SET PesoInChilogrammi = 56
WHERE ID = 0 (senza questo ogni record della tabella vengono modificati)
```

serve per cancellare da una tabella un record specifico

```
DELETE FROM Studente WHERE Cognome = 'Navarette'
```

aggiunge una colonna ad una tabella

```
ALTER TABLE Studente
ADD Telefono int

ALTER TABLE Studente
ALTER COLUMN Telefono varchar(10)
```

```
CREATE TABLE Studente(
        ID int IDENTITY(0,1) PRIMARY KEY,
        Cognome varchar(50),
        Nome varchar(50),

)

CREATE TABLE Telefono(
                                (indica che il valore è primario,o chiave
principale)
        ID int IDENTITY(0,1) PRIMARY KEY,
        Numero varchar(50),

        IDS int FOREIGN KEY REFERENCES Studente(ID)

)
```

COMANDI SQL

DATA DEFINITION LANGUAGE (schema/struttura)
-create
-alter
-drop

DATA MANIPULATION LANGUAGE(contenuto)
-delete
-insert into
-update

Si vuole creare un DB per inserire i voti degli studenti che prendono nelle varie materie. Di ogni studente si vogliono memorizzare nome cognome classe, delle materie siamo interessati al nome e al professore(nome cognome) ed infine per il voto siamo interessati al suo valore, alla data in cui è stato ricevuto e al tipo(scritto, orale o pratico).
Una volta realizzato il DB,aggiungere il campo anno scolastico al voto e inserire 2 studenti, 2 materie e 4 voti, 2 per ogni materia.
Scrivere poi le istruzioni per aggiungere 1 voto a tutti i voti nel senso che se uno ha preso 7 aggiungo 1 e diventerà 8 in questo senso.

Scrivere infine le istruzioni per eliminare tutti gli studenti dalla faccia della terra.

```
SELECT Progetto.Nome, Dipendente.Nome,Dipendente.Cognome
FROM Partecipa
INNER JOIN Progetto
on Partecipa.IDP = Progetto.ID
inner join Dipendente
on Partecipa.IDD = Dipendente.ID
WHERE Progetto.Nome = 'ING' (specifica quale informazione prendere,senza questo prende tutte le informazioni)
order by Dipendente.Nome
```

```
SELECT Progetto.Nome, Dipendente.Nome,Dipendente.Cognome
FROM Partecipa
INNER JOIN Progetto
on Partecipa.IDP = Progetto.ID
inner join Dipendente
on Partecipa.IDD = Dipendente.ID
WHERE Progetto.Budget BETWEEN 100000.00 AND 150000.00
order by Dipendente.Nome
```

LIKE

quando si vuole scrivere una query i passaggi sono i seguenti
1 individuare in quale tabelle ci sono le informazioni,se le informazioni sono su più tabelle devo fare una join individuando i campi che hanno in relazione
2 visualizzare i campi richiesti dal testo all'interno della **select**(selezione verticale,prendo solo alcuni campi)
3 se non sono necessari tutti i record ma solo alcuni,utilizzare la clausola **where**(selezione orizzontale,prendo alcuni record o righe della tabella)

il where con il group by non lo posso utilizzare,quindi si usa having

```
select model_year,count(*) as totale_prodotti from production.products
group by model_year
having count(*) >= 50
order by totale_prodotti desc
```

nel select si possono mettere i campi contenuti nel group by

```
select state,city,count(*) as clienti
from sales.customers
group by state,city
having count(*) >= 10
order by clienti desc
```

– somma del prezzo(maggiori di 50000) dei prodotti per ogni categoria

```
select category_name,sum(list_price) as somma_totale
from production.products inner join  production.categories
on production.products.category_id =
production.categories.category_id
group by products.category_id,category_name
having sum(list_price)>= 50000
order by sum(list_price) desc
```

--il prezzo del prodotto con il prezzo + alto (select nidificati)

```
select production.products.product_name
from production.products
where list_price =
(select max(list_price) as prezzo_massimo from
production.products)
```

1 individuazione dell'entità
2 individuazione degli attributi
3 verifico se degli attributi contengono + informazioni
4 relazione
5 cardinalità