Data la tabella:
Clienti (IdCliente, Cognome, Nome, Città, Salario,Eta,capo)

CREATE TABLE Cliente(
IdCliente INT AUTO_INCREMENT PRIMARY KEY,
Nome VARCHAR(50),
Cognome VARCHAR(50),
Citta VARCHAR(50),
Salario INT,
Eta INT
FOREIGN KEY(capo) REFERENCES azienda(capo)
); il punto e virgola determina la fine di un istruzione

SELECT Nome, Cognome
FROM Cliente
WHERE Salario>3000;

SELECT Cognome, Nome
FROM Cliente
WHERE Citta = “Rimini”;

SELECT Cognome, Nome
FROM Cliente
WHERE Citta=“Rimini” AND Salario>3000;

SELECT Cognome,Nome,Salario
FROM Cliente
WHERE Eta>20 AND Eta<=35;

Operazioni di confronto: <, >, <=, >=, =

Operatori logici
AND prodotto logico
OR somma logica
NOT negazione

DISCIPLINA

ID	NAME	ISMAN	DISTANCE
1	Men’s 100m	true	100m
2.	Men’s 200m	true	200m
3	Men’s 400m	true	400m
4	Men’s 800m	true	800m
5	Men’s 1500m	true	1500m

CREATE TABLE Disciplina(
Id INT AUTO_INCREMENT PRIMARY KEY,
NAME VARCHAR(50),
ISMAN BIT(1), valore buleano 1= true 0= false
DISTANCE INT
);

SELECT*
FROM DISCIPLINA

Supponi di dover progettare un sistema per gestire una biblioteca. La biblioteca deve tenere traccia dei libri, del lettori e dei prestiti effettuati. Ogni libro ha un titolo, un autore, e un codice isbn univoco. Ogni lettore ha un nome, un cognome e un numero di tessera univoco. Inoltre, è necessario memorizzare la data di inizio e di fine di ciascun prestito.

Biblioteca= idbiblioteca, indirizzo,
libro= titolo, autore, codice isbn univoco
lettore= nome, cognome, numero di tessera univoco

Funzioni di campo calcolato
MAX,MIN,AVG,SUM,COUNT

GROUP BY mi permette di creare gruppi di dati
Andrea 6 voti- Riccardo 3 voti- Tommaso 5 voti
SELECT COUNT (Voto.Studenteid) as Nvoti
FROM Studente,voto;
GROUP BY Studente.nome

```
SELECT AVG(Salario) AS Salariomedio
FROM Cliente;

SELECT Nome,Cognome,Eta
FROM Cliente
WHERE Salario>(SELECT AVG(Salario) FROM Cliente);

1 SELECT *
FROM Proprietari

2 SELECT nome, cognome
FROM Proprietari

3 SELECT *
FROM auto
WHERE colore = “rosso”
4 SELECT targa
FROM auto

5 SELECT auto
FROM auto
WHERE potenza>100

6 SELECT potenza
FROM auto
WHERE potenza>(SELECT AVG potenza AS potenza_med)

6 SELECT MAX potenza AS max_potenza
FROM auto

7 SELECT AVG potenza AS potenza_med
  FROM auto

8 SELECT AVG eta AS eta_med
  From proprietari
```

```
<?php
   echo”

            <form action=\”#\” method=\”GET\”>
      <input type=\”text\” name=\”a\” required/>
      <input type=\”submit\” value=\”calcola\”/>
   </form>
   “;

  $x=$_GET[‘a’];
  $p=1;
  for($i=2;$i<=$x;$i++)
       $p=$p*$i;

  echo”Il fattoriale di $x &egrave; $p”;
?>
```

Rivedere il codice completo di php e sopratutto la variabile asset a cosa serve e come si utilizza e il suo funzionamento

header(); <—— refresh della pagina automatico

Dovrò passare l’url della pagina che intendo caricare al termine del refresh

refresh <—- specifica dopo quanti secondi
url <—- specifica che cosa visualizzare

```
<?php
$a=5;
$b=4;
$c=-3;
   if($a>$b)
   if($a>$c)
            Echo”il numero &egrave:$a”;
   Else
```

```
            Echo” il numero &egrave:&c”;
   Else

    if($b>$c)
          Echo”il numero &egrave;$b”;
     Else
          Echo”il numero &egrave;$c”;
     Echo”fine programma”;
?>

<?php
$v=isset($_GET[‘v’])?$_GET[‘v’]:NULL;
$g=isset($_GET[‘g’])?$_GET[‘g’]:NULL;
if($v==NULL || $g==NULL)
Echo”
<form action = ‘esercizio verifica.php’ method=“GET”>
<input type = “text” name =“v” required>
<input type = “text” name= “g” required>
<input type =“submit” value =“invia”>
</form>
“;
Else
    for($i=0;$i<5;$i++){
        Echo”buongiorno $v $g!”;
}
?>
```

MODELLO LOGICO 1 ———— M
Artista 1 ————————————-|
Nome |
Cognome |
Data di nascita |
Nazionalità |
|
|
Quadro |

Titolo |
Tecnica |
Descrizione |
data_P |
cognomeArtista M ————————

Zona	Casa	Comprata	Cliente
Nome 1	**ID casa 1**	**M IDcasa**	**1IDcli**
Foto	DataC	dataV	
	Desc	data acq	
	M NomeZ	**IDcliente M**	

Studente
Matricola 1 2 3 <—il vettore scorre tutti i record con gli elementi di ogni colonna

```
SELECT *
FROM Studente
WHERE classe =\"  /"; <—tramite form
```

Studenti (nome,cognome, genere, classe)
materie(codmat, nome)
voto(data, tipo, voto, materiaid,studenteid)
L'entità voto crea la relazione molti a molti tra studente e materia

Firma elettronica= ha validità sono in quel contesto
Firma digitale= ha validità in tutto ciò che si fa
Pec= ente certificato che garantisce che quello che viene mandato e aperto è tutto vero. Va spedita da pec a pec

```
SELECT Matricola,Nome,Cognome,DataN,Classe
FROM Studente,Voto
WHERE Studente.Matricola=Voto.Studenteid AND Voto<6
//questo tipo di join viene definito  join di equivalenza
```

```
SELECT Studente.Nome,COUNT(Voto.Studenteid) as Nvoti
FROM Studente, Voto
WHERE Studente.Matricola=Voto.Studenteid
GROUP BY Studente.Nome;
```

```
1)SELECT marca, targa
  FROM auto
  WHERE Cilindrata>2000 OR potenza>120
2)SELECT proprietario.nome, targa
FROM proprietario, auto
WHERE proprietario.codf=auto.codf cilindrata>2000 OR
potenza>120
3)SELECT targa, proprietario.nome
FROM proprietario INNER JOIN(assicurazione INNER JOIN auto
ON assicurazione.ass=auto.cod.ass) ON proprietario.codf=
auto.codf
```

WHERE(cilindrata>2000 OR potenza>120) AND
assicurazione.nome=“sara”

Select attrazione.descrizione, count (giro.int) as num.giri
From attrazione inner join giro on attrazione.nattrazione=giro.nt
Where giro.datagiro = [ ]
Group by attrazione.descrizione;

Select attrazione.descrizione,count(giro.nt) as visite
From attrazione, giro, visitatore
Where attrazione.nattrazione = giro.na AND visitatore.ntessera = giro.nt AND attrazione.vietato = TRUE
Group by

Scrivere query che restituisce città e stipendio massimo di ogni città la cui età media sia inferiore a 35 anni
SELECT citta, MAX(salario) AS StipMax
FROM cliente
GROUP BY citta
HAVING AVG(eta)<35

Cosa non è un computer quantistico:
-Non è semplicemente un computer tradizionale, è molto più veloce e non è proprio digitale.
-non calcola tutte le possibili soluzioni contemporaneamente.

La meccanica quantistica permette ad un’entità di essere contemporaneamente in due o più stati.

Più importante è l’interferenza quantistica che permette di combinare dei “cammini” o ”alberi” computazionali:
-quelli che porterebbero a risposte sbagliate si cancellano tra di loro.
-quelli che portano le risposte giuste e si rafforzano.

Molti problemi possono essere formulati come una o più decisione da prendere:
-decidere se un certo numero naturale è primo
-decidere se una grandezza supera o no una certa soglia

Esistono algoritmi che utilizzano scelte casuali per velocizzare la ricerca di una soluzione:
-una sequenza di N numeri, per ordinare questi numeri si utilizza una scelta casuale che in media riduce il tempo.

Le transazioni sono regolate da ampiezze non dalle probabilità e le ampiezze possono essere sia positive che negative.

Nel caso quantistico si usa invece la norma2, definita come la somma dei quadrati.
-le coppie di numeri la cui somma dei quadrati è uguale a uno forma una circonferenza.

Cosa cambia nel caso quantistico?
Mentre non viene misurato, il qbit può essere in un punto qualsiasi della circonferenza;

L’ampiezza di arrivare in un particolare nodo (cioè una soluzione) è data dalla somma delle ampiezze dei cammini che partono dal primo nodo a sinistra ed arrivano a quel nodo

Nel calcolo quantistico si vuole che i cammini:
-Con ampiezza uguale a zero siano quelli che portano le risposte sbagliate.
-Che portano alle risposte giuste abbiano ampiezze che si combinano in maniera costruttiva(tutti positivi o tutti negativi).

Se si riesce a manipolare le ampiezze in modo da aumentare le possibilità di arrivare ad una risposta corretta, si velocizza la risoluzione del problema.
-il vantaggio che si ottiene dipende dal problema.

Neanche i dispositivi che implementano il calcolo classico digitale sono permetti ma hanno due vantaggi rispetto ai computer quantistici:
-un piccolo errore è facilmente individuabile

1)Migliorare la qualità dei qbit in varie tecnologie:
-atomi neutrali;
-punti quantistici;
-superconduttori ;
-ioni intrappolati;
-fotonica;

2)Definire ed implementare nuove tecniche per la rilevazione

Altri modi di sfruttare le proprietà della meccanica quantistica:
-Distribuire chiavi crittografiche in modo sicuro
-risolvere problemi di ottimizzazione sfruttando il principio del “quantum tunneling”

È possibile trovare associazioni di una stessa entità

```
SELECT Film.Titolo,SUM(Interpreta.Conpenso) AS CostoFilm
FROM Film INNER JOIN Interpreta ON
Film.IdFilm=Interpreta.IdFilm
WHERE Film.Nazionalita=“USA”
GROUP BY Film.Titolo
HAVING SUM(Interpreta.Conpenso)>20000;
```

```
SELECT Prodotti.nome, Disponibilita.num pezzi
FROM Prodotti INNER JOIN(Negozi INNER JOIN Disponibilita ON
negozi.id = Disponibilità.negozio)
```

```
SELECT  Negozio.nome, SUM(Disponibilita.num_pezzi)
FROM (Categorie IINER JOIN Prodotti ON categoria.id=
prodotti.categoria) INNER JOIN disponibilita ON
Prodotti.codice=disponibilita.prodotto
WHERE categoria.nome=“ufficio”
GROUP BY negozio.nome;
```

```
Cliente(codcliente,Nome,Cognome,Citta,Salario,eta)
Albergo(Cod alb,Nome,Citta)
Prenot(codcliente,codalbergo,Ngiorni,Acconto)

CREATE TABLE Cliente(
Codcliente INT AUTO_INCREMENT PRIMARY KEY
Nome VARCHAR(50)
Cognome VARCHAR(50)
Citta VARCHAR(50)
Salario INT
DataNascita DATE
);

CREATE TABLE Albergo(
Codalbergo INT AUTO_INCREMENT PRIMARY KEY
Nome VARCHAR(50)
Citta VARCHAR(50)
);
```

```
CREATE TABLE Prenot(
Dataprenot DATE
Acconto INT
Codcliente INT
Codalbergo INT
FOREIGN KEY(codcliente) REFERENCES Cliente(codcliente)
FOREIGN KEY (codalbergo) REFERENCES Albergo(codalbergo)
);
```

```
5)SELECT Cliente.nome, Cliente.Cognome,Cliente.citta
FROM Cliente INNER JOIN(albergo INNER JOIN prenot ON
albergo.cod_alb=prenot.cod_alb)ON cliente.cod_cli=prenot.cod_cli
WHERE pronto.acconto>(select avg(prenot.acconto)FROM prenot);
```

Si intende realizzare un database per raccogliere le migliori ricette stellate, in particolare per ogni ricetta si vuol tenere traccia dell'elenco degli ingredienti da usare. Nello specifico per ogni ingrediente si deve indicare la quantità da usare.Inoltre ogni ricetta può avere degli utensili, specifici che si dovrebbero usare per ottenere un buon risultato.Di ogni utensile va riportato il nome, la descrizione ed il sito dove poterlo acquistare. Le ricette sono ideate da chef diversi di cui si vuol memorizzare il nome, il cognome e il ristorante dove stanno lavorando attualmente. Di ogni ristorante si è interessati a conoscere l'indirizzo, il nome, numero di telefono, sito web e la descrizione

```
CREATE TABLE Musei(
NomeM VARCHAR(50) PRIMARY KEY
Citta VARCHAR(50)
);
```

```
CREATE TABLE Artisti(
NomeA VARCHAR(50) PRIMARY KEY
Nazionalità VARCHAR(50)
```

```
);
CREATE TABLE Opere(
INT Codice AUTO_INCREMENT PRIMARY KEY
Titolo VARCHAR(50)
NomeM VARCHAR(50)
NomeA VARCHAR(50)
FOREIGN KEY(NomeM) REFERENCES Musei(NomeM) ON
UPDATE CASCADE ON DELETE CASCADE
FOREIGN KEY(NomeA) REFERENCES Artisti(NomeA) ON
UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE Personaggi(
Personaggio VARCHAR(50) PRIMARY KEY
Codice INT
FOREIGN KEY(Codice) REFERENCES Opere(Codice)ON
UPDATE CASCADE ON DELETE CASCADE
);


1)SELECT codice, titolo
  FROM Opere
  WHERE NomeA=“Tiziano” AND NomeM=“National
Gallery”;

2)SELECT NomeA,Titolo
FROM Opere
WHERE NomeM=“Galleria degli Uffizi” OR NomeM=“National
Gallery”

3)SELECT NomeA,Titolo
FROM Musei INNER JOIN Opere ON
Museo.NomeM=Opere.NomeM
WHERE Museo.citta=“Firenze”

4)SELECT Museo.citta
FROM Museo INNER JOIN Opere ON
Museo.NomeM=Opere.NomeM
```

WHERE opere.NomeA=“Caravaggio”
GROUP BY Museo.citta

8)SELECT opere.Nome, COUNT(Titolo) As NumOpere
FROM Artisti INNER JOIN (Musei INNER JOIN opere ON Musei.NomeM=Opere.NomeM) ON Artisti.NomeA=Opere.NomeA
WHERE Museo.Citta=“LONDRA” AND Artista.Nazionalita=“Italiano”
GROUP BY Museo.NomeM

9)SELECT Museo.NomeM
FROM Opere
WHERE Opere.NomeA<>”Tiziano”AND Musei.Citta=“Londra”;

SELECT Opere.NomeM
FROM Musei
WHERE Musei.Citta=“Londra” AND “Tiziano” NOT IN (SELECT Opere.NomeA FROM Opere WHERE Musei.NomeM=Opere.NomeM)

7) SELECT Cliente.Nome,Cliente.Cognome,Cliente.Citta
FROM Cliente INNER JOIN(Prenot INNER JOIN albergo ON print.CodAlb=Albergo.CodAlb) ON prenot.CodCli=Cliente.CodCli
WHERE pronto.acconto>(SELECT AVG(Prenot.acconto) FROM prenot)

Per ogni artista visualizzare il numero di brani
SELECT COUNT(Brani.IdBrano) AS NumBrani,Artisti.Nome
FROM Artisti INNER JOIN Brani ON Artisti.IdArtista=Brani.IdArtista
GROUP BY Artisti.Nome
HAVING COUNT(Brani.IdBrano)>200

SELECT Brani.Genere
FROM Brani

Visualizzare il nome dell’artista con il maggior numero di brani

HAVING= permette di porre una condizione”scegliere come visualizzare” sui dati estratti oggetto di raggruppamento, questa funzione si può utilizzare sempre e solo dopo la funzione di Group by

GROUP BY= RAGGRUPPA le righe con lo stesso valore in un unico gruppo

ORDER BY ordinamento crescente e decrescente
IN= controlla se un valore appartiene a un insieme precisato NOT IN = controlla se un valore non appartiene a un insieme di valori
LIKE = confronta una stringa di caratteri simili con quella specificata

AS = funzione che mi permette di dare un soprannome o Alias ad un campo o entità

Applicazione	applicazione con il suo protocollo
Presentazione	(FTP,HTTP,SMTP)
Sessione	
	TCP/IP
Trasporto	protocollo TCP,UDP,ICMP
Rete	protocollo IP
Collegamento dati	protocollo della rete fisica sottostante
Fisico	

PTE=punto terminazione edificio

<form action=“estrazione.php” method=“GET”>

Dato un corso visualizzare quali sono i dati relativi ad esso e per quali anni scolastici è stato attivato

```
SELECT  Corso*
FROM Corso
WHERE Corso.CodiceCorso=[corso.titolo]
```

Per ogni anno scolastico, contare il numero di studenti respinti

```
SELECT Frequenza.AnnoFrequenza,COUNT(Matricola) AS
NumeroStudenti
FROM Frequenza
WHERE Frequenza.esito=“Non Promosso”
GROUP BY Frequenza.AnnoFrequenza
```

Autore
SCRIVE (PUBBLICA) Editore (CONTIENE)
Parolachiave=Codice,descrizione
Libro=idLibro,titolo,autore,editore,anno pubblicazione
PRENOTA=dataF,DataI
Socio=IdSocio,nome,cognome,dataN,ntel,email

Bilancio è fatto da due fattori:
Stato patrimoniale=scrivibile in qualsiasi momento e rappresenta ciò che la società ha
Conto economico=scrivibile in qualsiasi momento ed è l'insieme delle entrate e uscite finanziarie

Indici di redditività misurano la capacità dell'azienda di generare profitto rispetto ai ricavi, al capitale investito e al patrimonio netto.

R.O.I.(Return on investment)-Redditività del capitale investito

ROI=Utile Operativo/Capitale Investito x 100

ROE(Return on equity) Redditività del capitale proprio
ROE=Utile Netto/Patrimonio netto x 100

Indici di liquidità misurano la capacità dell'azienda di far fronte ai pagamenti nel breve termine
ROS(Return on sales)
Current Ratio
Quick Ratio

SELECT Libro.Idlibro,Libro.titolo.libro.Annopubblicazione
Credimi quando ti dico che non avrei voluto arrivare a tanto

FROM libro INNER JOIN(autore INNER JOIN scrive ON autore.IdAutore=scrive.IdAutore)ON Libro.IdLibro=scrive.IdLibro

SELECT libro.isbn,libro.titolo
FROM socio, libro, prenotazione
WHERE (socio.idsocio=prenotazione.idsocio AND libro.isbn=prenotazione.idlibro) AND (socio.nome=[inserire nome] AND socio.cognome=[inserire cognome]) AND (prenotazione.dataf IS NULL AND prenotazione.datal IS NOT NULL);

SELECT libro.isbn, libro.titolo
FROM libro INNER JOIN contiene ON libro.isbn=contiene.libro

WHERE contiene.parolachiave=[inserire parola chiave]

SELECT codice scambio
FROM scambio
WHERE anno scambio= [“”]

Create view<Nomevista>(si cerca una nuova tabella virtuale)
(attributi che la definisce)
AS query che estrae valori e li carica negli attributi della vista logica

Per ogni codice sede i progetti attivi
Create view progetto sede(codprog,codice sede)
AS SELECT progetto comprogetto, s sede

SELECT Insegnamento.Nome, Telefono
FROM Insegnante
Order by Insegnamento.Nome ASC

SELECT Nome,Cognome(stipendio*1936,27)AS stipendio
FROM Insegnante
ORDER BY Nome ASC

SELECT Insegnante.Nome, Insegnante.Cognome
FROM Insegnante
WHERE (citta=“Milano” OR citta=“Verona”)AND Stipendio>2000;

SELECT Sum(stipendio) AS sommastipendi
FROM Insegnante
WHERE stipendio>2000

SELECT Studente.Matricola
FROM Studente INNER JOIN Esame ON
Studente.Matricola=Esame.MatricolaStudente
Where Esame.Voto=30 AND Esame.Voto=31;

SELECT Studente.nome,Studente.cognome,Studente.matricola

```
FROM Studenti INNER JOIN(corso INNER JOIN esame ON
corso.codice=esame.codice)ON
studente.matricola=esame.matricola
WHERE esame.voto=30 AND studente M/F=true;

SELECT COUNT(matricola.studente) AS Studenti
FROM Studente, esame, corso, insegnamento, insegnante
WHERE matricola.studente=esame.matricolastudente, esame.
codicecorso=codice.corso, corso.codice=
insegnamento.codicecorso, insegnamento.matricolaprofessore=
insegnante.matricola AND corso.nome= “base di dati” AND
insegnamento.nome=“letizia” AND insegnante.cognome=“tanca”
AND esame.voto>=18;

SELECTnome,cognome,matricola,COUNT(matricolastudente) AS
nStudente
FROM (insegnante INNER JOIN (corso INNER JOIN insegnamento
ON corso.codice=insegnamento.codicecorso) ON
insegnante.matricola=insegnamento.matricolaprofessore) AND
corso INNER JOIN esame ON corso.codice=esame.codicecorso


SELECT Attrazione.descrizione,COUNT(attrazione.nattrazione) AS
nvolteatttrazione
FROM attrazione INNER JOIN  giri ON
attrazione.nattrazione=girinattrazione
WHERE giri.data=[“inserisci una data”]
GROUP BY attrazione.descrizione

SELECT attrazione.descrizione, COUNT(giri.nattrazione) AS nvisite
FROM attrazione INNER JOIN giri ON
attrazione.nattrazione=giri.nattrazione
GROUP BY attrazione.descrizione;

SELECT nomea
FROM tabvisite
WHERE nvisite=[SELECT MAX(nvisite) FROM tabvisite]
```

```
SELECT IdAttore,COUNT(idFilm) AS nFilm
FROM Interpreta
GROUP BY IdAttore


SELECT Albergo.citta
FROM Albergo
GROUP BY Albergo.citta
HAVING COUNT(Cod_Alb)>=10;


CREATE TABLE Utente(
Idutente INT AUTO_INCREMENT PRIMARY KEY,
Nome VARCHAR(50);
Cognome VARCHAR(50);
DataN Date;
Sesso Bit;
Amministratore Bit;
Indirizzo VARCHAR(200)
);

CREATE TABLE Scheda(
IdQuadro INT AUTO_INCREMENT PRIMARY KEY,
Autore VARCHAR(50),
Titolo VARCHAR(50),
Immagine VARCHAR(50),
Prezzo DOUBLE,
Dimensione DOUBLE,
Tecnica,
ENUM('Olio','Carboncino',litografia','tempera')
);

CREATE TABLE consulta(
IdUtente INT,
FOREIGN KEY (utente) REFENCES utente(idutente) ON UPDATE
CASCADE ON DELETE CASCADE
IdQuadro INT,
FOREIGN KEY(idquadro) REFERENCES scheda(idquadro) ON
UPDATE CASCADE ON DELETE CASCADE
```

```
OraAccesso TIME,
DataAccesso DATE
);

SELECT Titolo,Autore,Immagine
FROM Scheda
WHERE  Tecnica=[“inserire una tecnica pittorica”]


SELECT Titolo,Autore,Immagine
FROM Scheda
WHERE Autore=[“Inserire un autore“] AND Prezzo<300;


sintagma=espressione



SELECT Q1
FROM Q1
WHERE CompensoTot=(SELECT MAX(CompensoTot) FROM Q1)




CREATE TABLE Cliente(
IdCliente INT AUTO_INCREMENT PRIMARY KEY
Cognome VARCHAR(50),
Nome VARCHAR(50),
Citta VARCHAR(50),
Salario INT,
Eta  INT
);

2)SELECT citta,COUNT(cognome) AS Nabitanti,AVG(eta)AS
etamedia
FROM cittadini
GROUP BY citta;
```

3)SELECT citta,AVG(eta) AS etamedia
FROM cittadini
GROUP BY citta
HAVING AVG(eta)>30;

SELECT MAX(Salario) AS Salario_massimo
FROM Cliente
WHERE citta='Rimini';

SELECT Nome, Cognome, Citta, eta
FROM Cliente
WHERE citta= ['Inserisci la citta']

SELECT insegnante.nome, insegnante.cognome, insegnante.matricola
FROM insegnante
WHERE stipendio=(SELECT MAX(Stipendio) AS stipendio_max
FROM insegnante)

SELECT nome, cognome
FROM insegnante
WHERE citta='milano' OR citta='verona' AND stipendio>2000;

SELECT SUM(stipendio)
FROM insegnante
WHERE stipendio>2000; oppure HAVING stipendio>2000 (perchè non sto lavorando su un gruppo, lo utilizzo come se fosse un where)

Artista, brano, genere, album

A. SELECT Autore.nome,SUM(dischi.ncopievendute) AS somma copie vendute

```
    FROM Autori INNER JOIN Dischi ON
Autori.CodAutore=Dischi.CodAutore
WHERE ncopievendute=[SELECT MAX(copie) AS
ncopievendute_mas FROM Dischi]
GROUP BY autore.nome
```

```
2) SELECT Film.Titolo,Film.Idfilm,SUM(proiezione.incasso) AS
incasso tot
FROM Film INNER JOIN proiezione ON
film.Idfilm=Proiezione.Idfilm
GROUP BY Film.idifilm,Film.titolo

SELECT Q1.titolo
FROM Q1
WHERE incassotot=(SELECT MAX(incassotot) FROM Q1);
```