# Tabelle e query SQL di base

## Tabella Cliente

```sql
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
```

## Query di base

```sql
-- Seleziona nome e cognome dei clienti con salario > 3000
SELECT Nome, Cognome
FROM Cliente
WHERE Salario > 3000;

-- Seleziona cognome e nome dei clienti di Rimini
SELECT Cognome, Nome
FROM Cliente
WHERE Citta = "Rimini";

-- Seleziona cognome e nome dei clienti di Rimini con salario > 3000
SELECT Cognome, Nome
FROM Cliente
WHERE Citta = "Rimini" AND Salario > 3000;

-- Seleziona cognome, nome e salario dei clienti con età tra 21 e 35 anni
SELECT Cognome, Nome, Salario
FROM Cliente
WHERE Eta > 20 AND Eta <= 35;
```

## Operatori

- Operazioni di confronto: `<`, `>`, `<=`, `>=`, `=`
- Operatori logici:
  - `AND` prodotto logico
  - `OR` somma logica
  - `NOT` negazione

## Tabella Disciplina

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

SELECT *
FROM DISCIPLINA;
```

## Esercizio progettazione biblioteca

Progettare un sistema per gestire una biblioteca:

- Biblioteca = idbiblioteca, indirizzo
- Libro = titolo, autore, codice isbn univoco
- Lettore = nome, cognome, numero di tessera univoco
- Prestito = data inizio, data fine

## Funzioni aggregate e GROUP BY

```sql
-- Funzioni di campo calcolato: MAX, MIN, AVG, SUM, COUNT

-- GROUP BY: crea gruppi di dati
-- Esempio: Andrea 6 voti, Riccardo 3 voti, Tommaso 5 voti
SELECT COUNT(Voto.Studenteid) AS Nvoti
FROM Studente, Voto
GROUP BY Studente.nome;

-- Calcola il salario medio
SELECT AVG(Salario) AS Salariomedio
FROM Cliente;

-- Seleziona dati dei clienti con salario superiore alla media
SELECT Nome, Cognome, Eta
FROM Cliente
WHERE Salario > (SELECT AVG(Salario) FROM Cliente);
```

## Esempi di query con auto e proprietari

```sql
SELECT *
FROM Proprietari;

SELECT nome, cognome
```

```sql
FROM Proprietari;

SELECT *
FROM auto
WHERE colore = "rosso";

SELECT targa
FROM auto;

SELECT auto
FROM auto
WHERE potenza > 100;

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

SELECT MAX(potenza) AS max_potenza
FROM auto;

SELECT AVG(potenza) AS potenza_med
FROM auto;

SELECT AVG(eta) AS eta_med
FROM proprietari;
```

# PHP e HTML

## Esempio calcolo fattoriale

```php
<?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";
?>
```

## Note su funzioni PHP

- `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

## Altri esempi PHP

```php
<?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
<?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 e relazioni

## Relazione 1:M

```
Artista (1) ------- (M) Quadro
Nome                    Titolo
Cognome                 Tecnica
Data di nascita         Descrizione
Nazionalità             data_P
                        cognomeArtista
```

## Altre relazioni

```
Zona (1) --- Casa (1) --- Comprata (M) --- Cliente
Nome         ID casa       IDcasa           IDcli
Foto         DataC         dataV
              Desc         data acq
             NomeZ         IDcliente
```

# Esempio di modello con più entità

```
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

# Note su firme elettroniche

- Firma elettronica = ha validità solo 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

# JOIN in SQL

```sql
-- Join di equivalenza
SELECT Matricola, Nome, Cognome, DataN, Classe
FROM Studente, Voto
WHERE Studente.Matricola = Voto.Studenteid AND Voto < 6;

-- Esempio di join con conteggio
SELECT Studente.Nome, COUNT(Voto.Studenteid) as Nvoti
FROM Studente, Voto
WHERE Studente.Matricola = Voto.Studenteid
GROUP BY Studente.Nome;
```

## Esempi di query complesse

```sql
-- Query con OR
SELECT marca, targa
FROM auto
WHERE Cilindrata > 2000 OR potenza > 120;

-- Join multipli con proprietario
SELECT proprietario.nome, targa
FROM proprietario, auto
WHERE proprietario.codf = auto.codf AND (cilindrata > 2000 OR potenza >
120);

-- Join annidati
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";

-- Query con GROUP BY e aggregazioni
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;

-- Query con HAVING
SELECT citta, MAX(salario) AS StipMax
FROM cliente
GROUP BY citta
HAVING AVG(eta) < 35;
```

# Computer quantistici

## Concetti base

- Non è semplicemente un computer tradizionale, è molto più veloce e non è propriamente digitale
- Non calcola tutte le possibili soluzioni contemporaneamente
- La meccanica quantistica permette ad un'entità di essere contemporaneamente in due o più stati

## Interferenza quantistica

- Permette di combinare dei "cammini" o "alberi" computazionali
- Quelli che porterebbero a risposte sbagliate si cancellano tra di loro
- Quelli che portano alle risposte giuste si rafforzano

## Algoritmi e problemi

- Molti problemi possono essere formulati come decisioni da prendere
- Esistono algoritmi che utilizzano scelte casuali per velocizzare la ricerca
- Le transazioni sono regolate da ampiezze non dalle probabilità
- Le ampiezze possono essere sia positive che negative
- Nel caso quantistico si usa la norma2, definita come la somma dei quadrati

## Qbit

- Mentre non viene misurato, il qbit può essere in un punto qualsiasi della circonferenza
- L'ampiezza di arrivare in un nodo è data dalla somma delle ampiezze dei cammini

## Sfide e tecnologie

- Migliorare la qualità dei qbit in varie tecnologie:
  - Atomi neutrali
  - Punti quantistici
  - Superconduttori
  - Ioni intrappolati
  - Fotonica
- Definire ed implementare nuove tecniche per la rilevazione

## Applicazioni

- Distribuire chiavi crittografiche in modo sicuro
- Risolvere problemi di ottimizzazione sfruttando il principio del "quantum tunneling"

# Altri esempi di query SQL

```sql
-- Query con JOIN e SUM
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;

-- Join multipli
SELECT Prodotti.nome, Disponibilita.num_pezzi
FROM Prodotti INNER JOIN (Negozi INNER JOIN Disponibilita ON negozi.id =
Disponibilità.negozio);

-- Query con SUM e GROUP BY
SELECT Negozio.nome, SUM(Disponibilita.num_pezzi)
FROM (Categorie INNER JOIN Prodotti ON categoria.id = prodotti.categoria)
```

```
INNER JOIN disponibilita ON Prodotti.codice = disponibilita.prodotto
WHERE categoria.nome = "ufficio"
GROUP BY negozio.nome;
```

## Schema database hotel

```
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)
);
```

## Schema database museo

```
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(
    Codice INT 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
);
```

## Query su opere d'arte

```
-- Trova opere di Tiziano alla National Gallery
SELECT codice, titolo
FROM Opere
WHERE NomeA = "Tiziano" AND NomeM = "National Gallery";

-- Trova opere in due musei specifici
SELECT NomeA, Titolo
FROM Opere
WHERE NomeM = "Galleria degli Uffizi" OR NomeM = "National Gallery";

-- Trova opere nei musei di Firenze
SELECT NomeA, Titolo
FROM Musei INNER JOIN Opere ON Museo.NomeM = Opere.NomeM
WHERE Museo.citta = "Firenze";

-- Trova città con opere di Caravaggio
SELECT Museo.citta
FROM Museo INNER JOIN Opere ON Museo.NomeM = Opere.NomeM
WHERE opere.NomeA = "Caravaggio"
GROUP BY Museo.citta;

-- Conta opere italiane nei musei di Londra
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;

-- Trova musei di Londra senza opere di Tiziano
SELECT Opere.NomeM
FROM Musei
WHERE Musei.Citta = "Londra" AND "Tiziano" NOT IN (SELECT Opere.NomeA FROM
Opere WHERE Musei.NomeM = Opere.NomeM);
```

# Comandi SQL aggiuntivi

- `HAVING` = permette di porre una condizione sui dati estratti dopo il raggruppamento. Si utilizza sempre dopo 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. `NOT IN` = controlla se non appartiene
- `LIKE` = confronta una stringa di caratteri simili con quella specificata
- `AS` = dà un soprannome o alias a un campo o entità

# Schema stack di rete

```
Applicazione          (FTP, HTTP, SMTP)
Presentazione
Sessione              TCP/IP
Trasporto             (protocollo TCP, UDP, ICMP)
Rete                  (protocollo IP)
Collegamento dati     (protocollo della rete fisica)
Fisico
```

# PTE = Punto Terminazione Edificio

# Indici di redditività aziendale

- 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

# Schema database galleria d'arte

```sql
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 (IdUtente) REFERENCES 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
);
```