5) Visualizzare quante specie diverse di pino sono presenti in ciascun parco.

```
SELECT Flora.Albero, Parco.Nome, COUNT(*) AS Numero_specie
FROM Parco
INNER JOIN Presente ON Parco.Nome = Presente.Nome
INNER JOIN Flora ON Flora.Codice_fl = Presente.Codice_fl
WHERE Tipo = "Pino"
GROUP BY Flora.Albero, Parco.Nome;
```

Considerando "diverse" (self-join: cosa avanzata)

```
SELECT Flora.Albero, Parco.Nome, COUNT(*) AS Numero_specie
FROM Parco P1, Parco P2
INNER JOIN Presente ON P1.Nome = Presente.Nome
INNER JOIN Flora ON Flora.Codice_fl = Presente.Codice_fl
INNER JOIN Parco P2
WHERE Tipo = "Pino"
AND P1.Nome <> P2.Nome
GROUP BY Flora.Albero, Parco.Nome;
```

6) Calcolare il numero medio dei cuccioli di ciascuna specie presenti in tutti i parchi della regione.

```
(SELECT COUNT(Fauna.Specie) AS Numero, Parco.Nome, Fauna.Specie
FROM Fauna
INNER JOIN Vive ON Fauna.Codice_fauna = Vive.Codice_fauna
INNER JOIN Parco ON Parco.Nome = Vive.Nome
WHERE Esemplare.Data_Nascita < "2007-06-17"
GROUP BY Parco.Nome, Fauna.Specie) AS Q1;

SELECT Q1.Nome, Q1.Specie
FROM Q1
WHERE Q1.Numero = (SELECT AVG(Numero) FROM Q1);
```

7) Visualizzare l'esemplare più anziano di ogni specie presente in un determinato parco

```
SELECT Esemplare.Codice_e, Fauna.Specie, MAX(Data_nascita) AS
Esemplare_piu_anziano
FROM Fauna
INNER JOIN Esemplare ON Fauna.Codice_fa = Esemplare.Codice_fa
WHERE Parco = "Sigurtà"
GROUP BY Esemplare.Codice_e, Fauna.Specie;
```

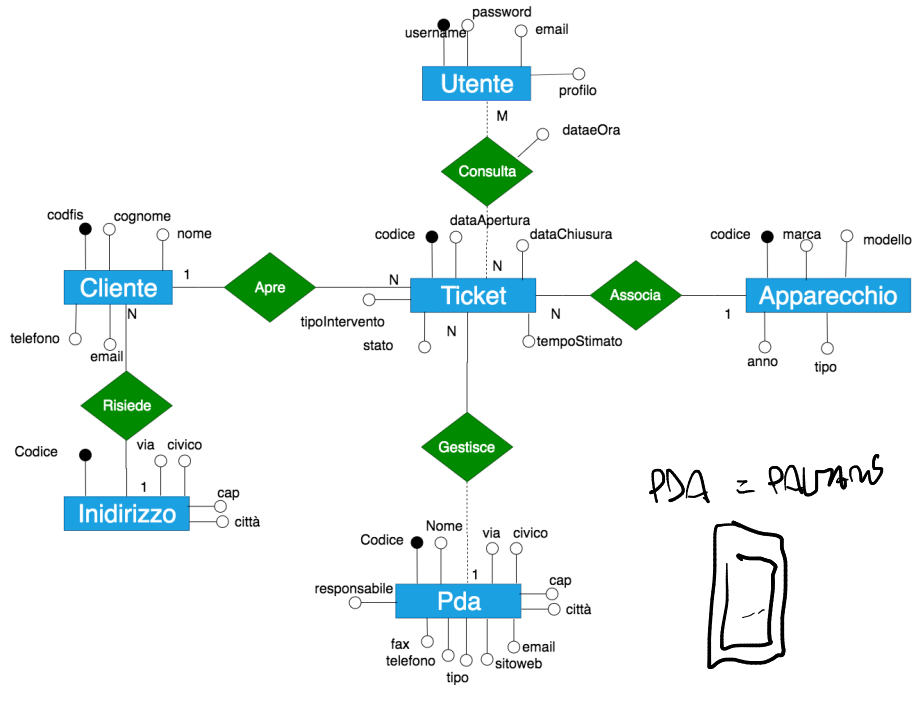
- La definizione delle relazioni e le seguenti interrogazioni espresse in linguaggio SQL:
  1. visualizzare l'elenco, in ordine cronologico, di tutti gli interventi di riparazione di telefoni cellulari del tipo "sostituzione display" effettuati nell'arco di un mese;
  2. visualizzare i dati dei clienti ai quali, dopo trenta giorni, non è stato ancora riparato l'articolo consegnato;
  3. data una marca ed un modello, visualizzare la durata media degli interventi di riparazione per i prodotti di tale marca e modello;
  4. calcolare e visualizzare quanti interventi di riparazione, andati a buon fine, sono stati effettuati, suddivisi per marca, nell'arco di un anno;
  5. calcolare e visualizzare quanti TICKET sono stati compilati da ciascun PDA e la durata media di lavorazione dei TICKET;
  6. dato il codice identificativo di un intervento, visualizzarne lo stato.

1. Visualizzare l'elenco, in ordine cronologico, di tutti gli interventi di riparazione di telefoni cellulari del tipo "sostituzione display" effettuati nell'arco di un mese.

```
SELECT Ticket.TipoIntervento
FROM Ticket
WHERE Ticket.TipoIntervento = "Sostituzione display"
AND Ticket.DataChiusura BETWEEN (["2025-06-01"],
["2025-06-30"])
ORDER BY Ticket.DataChiusura;
```

2. Visualizzare i dati dei clienti ai quali, dopo trenta giorni, non è stato ancora riparato l'articolo consegnato;

```
SELECT Cliente.CodFis, Cliente.Nome
FROM Cliente
INNER JOIN Ticket ON Cliente.Ticket = Ticket.Codice
WHERE [NOW() - Ticket.DataApertura] > 30
AND Ticket.DataChiusura IS NULL
AND Ticket.Stato = "In riparazione";
```



Q6: DATO IL CODICE IDENTIFICATIVO DI UN INTERVENTO, VISUALIZZARNE LO STATO

```
SELECT Stato
FROM Ticket
WHERE ( CodTicket = [ Codice Immesso ]);
```

~~$\pi_A(\sigma_P(Ticket))$ A = {Stato} P = {CodTicket = ...}~~

QUANDO HAI INPUT UTENTE → [INPUT IMMESSO]

3. Data una marca ed un modello,
visualizzare la durata media degli interventi di
riparazione per i prodotti di tale marca e modello

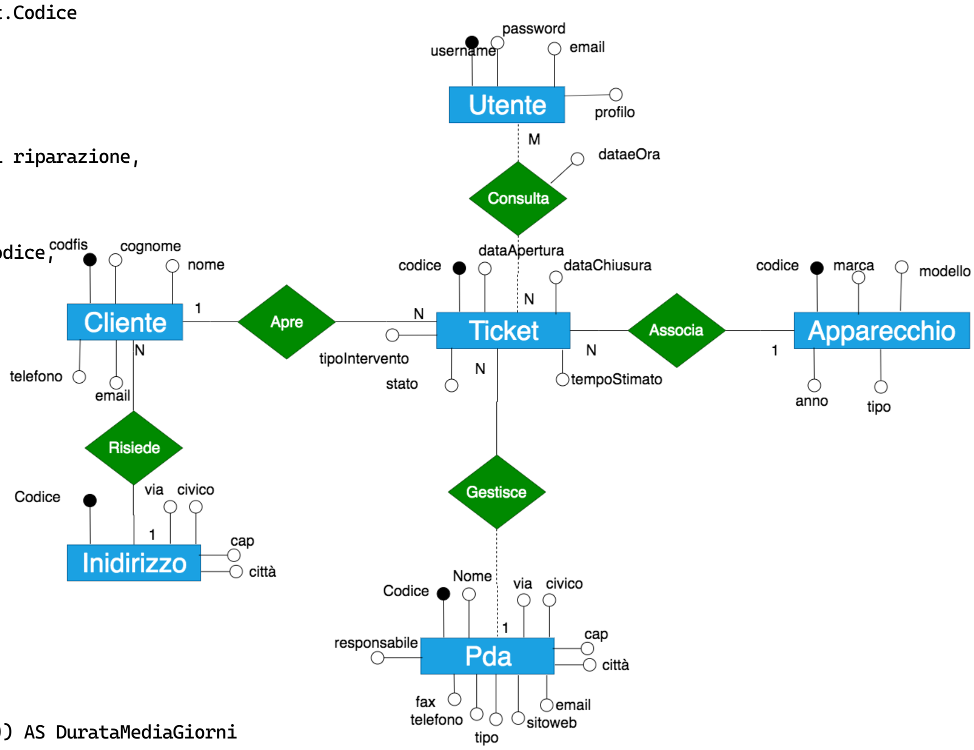
```
SELECT AVG(DataApertura - DataChiusura) AS Durata
FROM Apparecchio
INNER JOIN Ticket ON Apparecchio.Ticket =
Ticket.Codice
WHERE Apparecchio.Marca = []
AND Apparecchio.Modello = []
AND Ticket.TipoIntervento = "Riparazione"
```

Soluzione "utile"

```
SELECT AVG(DATEDIFF(DataChiusura, DataApertura))
AS DurataMedia
FROM Apparecchio
INNER JOIN Ticket ON Apparecchio.Ticket = Ticket.Codice
WHERE Apparecchio.Marca = 'X'
AND Apparecchio.Modello = 'Y'
AND Ticket.TipoIntervento = 'Riparazione'
AND Ticket.Stato = 'chiuso'
```

4. Calcolare e visualizzare quanti interventi di riparazione,
andati a buon fine, sono stati effettuati,
suddivisi per marca, nell'arco di un anno;

```
SELECT COUNT(Ticket.Codice) AS Numero, Ticket.Codice,
Apparecchio.Marca
FROM Ticket
JOIN Apparecchio ON Ticket.Codice =
Apparecchio.Ticket
WHERE Ticket.Stato = "Positivo"
AND Ticket.DataChiusura BETWEEN [("2025-01-01"),
("2025-12-31")]
GROUP BY Ticket.Codice, Apparecchio.Marca;
```



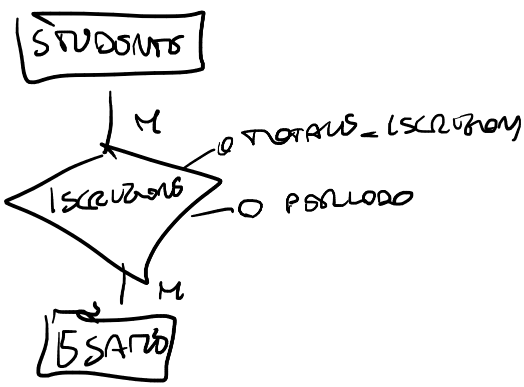
5. Ticket per PDA e durata media di lavorazione:

```
SELECT
    p.Codice AS CodicePDA,
    p.Nome AS NomePDA,
    COUNT(t.Codice) AS NumeroTicket,
    AVG(DATEDIFF(t.DataChiusura, t.DataApertura)) AS DurataMediaGiorni
FROM Pda p
LEFT JOIN Ticket t ON p.Codice = t.Pda
WHERE t.DataChiusura IS NOT NULL  -- solo ticket chiusi
GROUP BY p.Codice, p.Nome
ORDER BY NumeroTicket DESC
```

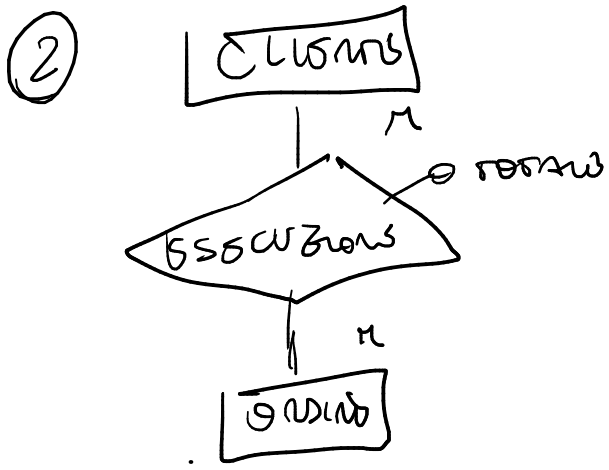
6. Stato di intervento dato il codice

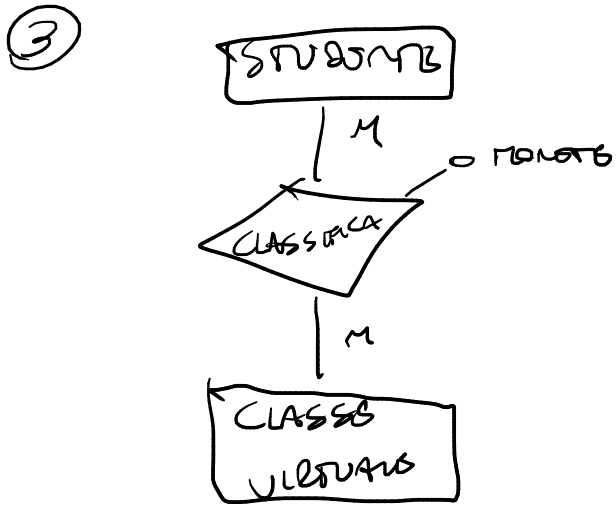
```
SELECT stato
FROM Ticket
WHERE codice = ?  -- parametro da sostituire
```

RELAZIONI / ATTRIBUTI



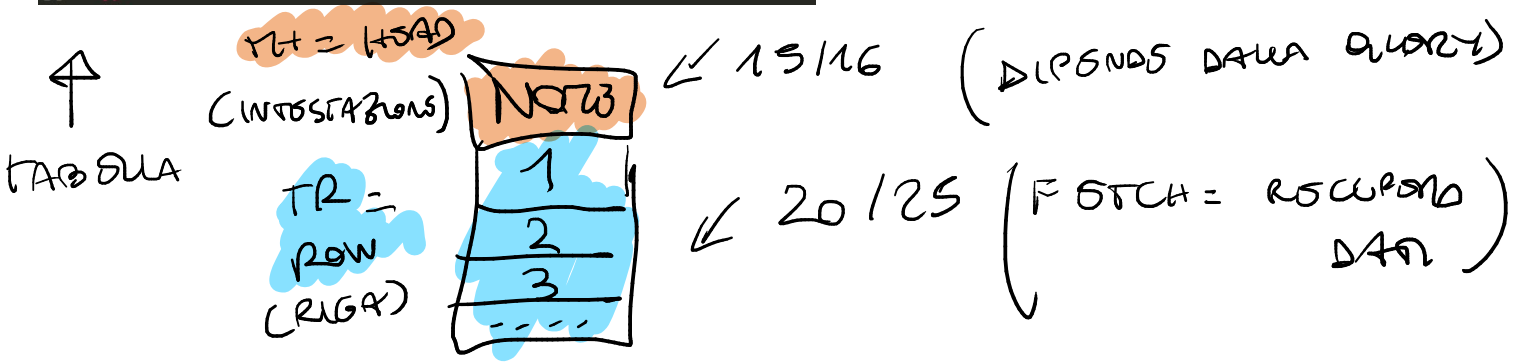
(DATI CHE COINVOLGONO GENERALMENTE LE ENTITÀ!)





```php
<?php
$conn = mysqli_connect('localhost','root','password','database');

if(!$conn) echo"errore";
else{
$query = "SELECT Nome
          FROM Tabella";
}

$result = mysqli_query($conn, $query);
if(!$result) echo"errore nella query";
else{
    echo"
        <table>
            <tr>
                <th>Nome<th>
            </tr>
        ";

        while($vett = mysqli_fetch_array($result,MYSQLI_ASSOC)){
            echo"
                <tr>
                <td>$vett[Nome]</td>
                </tr>";
        }

        echo"</table>";
}
mysqli_close($conn)
?>
```

→ ① CONNESSIONE
SE NO CONN → ERRORE
ALTRIMENTI
→ ② QUERY
SE NO → ERRORE
→ ③ ESECUZIONE E RECUPERO RISULTATI
→ ④ CHIUSURA CONNESSIONE



CODICE (IMMAGINE)
[UNA RIGA]

→ ULTIMI PUNTI PARTE 1
→ SECONDA PARTE

FUNZIONI SQL

- SELECT ⟶ Visualizzazione righe output
- FROM ⟶ Tabelle
- (INNER) JOIN ⟶ Relazioni tra tabelle
- WHERE ⟶ Condizioni

Funzioni secondarie:

- COUNT / AVG / MIN / MAX / SUM → Funzioni di aggregazione

- I campi NON già coinvolti in funzioni di aggregazione VANNO MESSI CON "GROUP BY"

- HAVING ⟶ Come il WHERE, ma per le funzioni di aggregazione!

Esempio:
```
SELECT COUNT(*) AS Numero
...
HAVING COUNT(*) ≥10
```

- IN / NOT IN ⟶ Usate con le sottoquery per assicurarti di prendere/ non prendere determinati dati

Esempio: prendere i nomi di chi si laurea nel 2025

(Prendo solo quelli che hanno come anno di laurea 2025)

```
SELECT Nome
FROM Studente
WHERE Matricola IN (SELECT Nome FROM Studente WHERE Anno_laurea = 2025);
```

OPPURE

(Solo quelli che non si laureano nel 2025 e li togli - nota: il simbolo <> significa "diverso"

```
SELECT Nome
FROM Studente
WHERE Matricola NOT IN (SELECT Nome FROM Studente WHERE Anno_laurea <> 2025);
```

Funzioni con le date:

- YEAR ⟶ Estrae l'anno da una data ("YYYY-MM-DD")
Es. ("2025-05-05") ⟶ 2025

- NOW() ⟶ Estrae la data corrente
- BETWEEN ⟶ Intervallo di date
BETWEEN["data inizio", "data fine"

CASI PARTICOLARI:

- Verificare che un conteggio sia DAVVERO la media o il massimo (o il minimo)

```
(SELECT COUNT(*) AS Numero FROM ....) AS Q1;

SELECT Numero FROM Q1 WHERE Numero = (SELECT MAX(Numero) FROM Q1);
```

- Il testo chiede "un dato immesso in input" oppure "un dato messo dall'utente"

```
WHERE Dato = []
```
OPPURE
(gli metti tu un valore di riferimento ⟶ chiedi al prof.)

```
WHERE Dato = "12345"
```