Esame di Stato Informatica - Approccio Operativo

---

1. COME LEGGERE E ANALIZZARE LA TRACCIA

Tecnica del "Markup Attivo"

Primo passaggio (5 minuti):

```
1. Evidenziare in GIALLO tutti i sostantivi → entità candidate
2. Sottolineare in ROSSO i verbi di azione → relazioni
3. Cerchiare in BLU i numeri e vincoli → attributi e cardinalità
```

Esempio pratico (traccia EasyTrain):

```
"La compagnia ferroviaria [EasyTrain] fornisce servizi di [viaggio]
a lunga percorrenza. L'utente può [prenotare] un viaggio,
selezionando [carrozza] e [posto]"

→ Entità: EasyTrain, Viaggio, Utente, Carrozza, Posto
→ Relazioni: prenotare, selezionare
→ Vincoli: "lunga percorrenza", cardinalità implicite
```

Strategia della "Domanda Inversa"

Per ogni query richiesta, formulare subito:

- "Quali tabelle mi servono per ottenere questi dati?"
- "Che JOIN devo fare?"
- "Ci sono aggregazioni?"

---

2. COSTRUZIONE SCHEMA E/R - METODO PRATICO

Tecnica "Entità-Prima"

Step 1: Disegnare rettangoli per tutte le entità evidenziate **Step 2:** Per ogni entità, scrivere 3-4 attributi ovvi **Step 3:** Collegare con rombi (partire dalle relazioni più semplici)

Pattern di Riconoscimento Immediato

Scenario utente-contenuto (90% delle tracce):

```
UTENTE ──(1)──── CREA ──(N)──── CONTENUTO
   │                              │
   │                              │
   └──(N)── VALUTA ──(M)──────────┘
```

Scenario gerarchico (città/regioni):

```
REGIONE ──(1)──── CONTIENE ──(N)──── PROVINCIA ──(1)──── CONTIENE ──(N)──── COMUNE
```

Trucco per le Cardinalità

Regola del "Chi può avere molti":

- Un utente può avere molti viaggi → (1:N)
- Un viaggio può avere molti passeggeri → (N:M) tramite prenotazione
- Un film può avere molti attori E viceversa → (N:M)

---

3. SCHEMA LOGICO - CONVERSIONE MECCANICA

Algoritmo Standard

```
PER OGNI entità:
    CREA tabella con nome_plurale
    AGGIUNGI id_autoincrement come PK
    COPIA tutti gli attributi

PER OGNI relazione 1:N:
    AGGIUNGI FK nella tabella "N"

PER OGNI relazione N:M:
    CREA tabella con nome_relazione
    AGGIUNGI FK di entrambe le entità
    PK composta dalle due FK
```

Esempio Meccanico (traccia film)

```
E/R: UTENTE (1:N) VISUALIZZA (N:M) FILM

Logico:
- utenti(id, nome, email)
- film(id, titolo, genere, anno)
- visualizzazioni(utente_id, film_id, data)
  PK(utente_id, film_id)
  FK utente_id → utenti(id)
  FK film_id → film(id)
```

---

4. SQL TABELLE - TEMPLATE STANDARD

Formula Base per ogni tabella

```sql
CREATE TABLE nome_tabella (
    id INT AUTO_INCREMENT PRIMARY KEY,
    -- attributi normali
    campo1 VARCHAR(100) NOT NULL,
    campo2 DATE,
    -- foreign keys sempre alla fine
    altra_tabella_id INT,
    FOREIGN KEY (altra_tabella_id) REFERENCES altra_tabella(id)
);
```

Tipi di Dati per Categoria

```sql
-- Identificatori
id INT AUTO_INCREMENT PRIMARY KEY

-- Testi
nome VARCHAR(50), descrizione TEXT, email VARCHAR(100)

-- Numeri
prezzo DECIMAL(10,2), quantita INT, valutazione TINYINT

-- Date
data_nascita DATE, timestamp_creazione DATETIME

-- Enumerazioni
stato ENUM('attivo','sospeso','eliminato')
```

---

5. QUERY SQL - RICETTE PRATICHE

Query Tipo 1: Elenco Semplice con JOIN

Pattern: "Elencare X con informazioni di Y"

```sql
-- Template
SELECT t1.campo1, t2.campo2
FROM tabella_principale t1
JOIN tabella_correlata t2 ON t1.fk_id = t2.id
WHERE condizione_filtro
ORDER BY campo_ordinamento;

-- Esempio pratico: autisti con auto per un viaggio
SELECT u.nome, u.cognome, a.modello, v.prezzo
FROM utenti u
JOIN auto a ON u.id = a.proprietario_id
JOIN viaggi v ON u.id = v.autista_id
WHERE v.citta_partenza = 'Milano'
ORDER BY v.ora_partenza;
```

Query Tipo 2: Conteggi e Medie

Pattern: "Numero di X per ogni Y" o "Media di X"

```sql
-- Template
SELECT campo_raggruppamento, COUNT(*) as totale
FROM tabella t1
JOIN altra_tabella t2 ON t1.fk = t2.id
WHERE filtro_opzionale
GROUP BY campo_raggruppamento
HAVING COUNT(*) > soglia_opzionale
ORDER BY totale DESC;

-- Esempio: film per genere con più di 10 visualizzazioni
SELECT f.genere, COUNT(*) as num_visualizzazioni
FROM film f
JOIN visualizzazioni v ON f.id = v.film_id
GROUP BY f.genere
HAVING COUNT(*) > 10
ORDER BY num_visualizzazioni DESC;
```

Query Tipo 3: Sottquery con NOT IN

Pattern: "Utenti che NON hanno mai fatto X"

```sql
-- Template
SELECT campo1, campo2
FROM tabella_principale
WHERE id NOT IN (
    SELECT DISTINCT fk_id
    FROM tabella_azioni
    WHERE fk_id IS NOT NULL
);

-- Esempio: utenti senza visualizzazioni
SELECT nome, cognome
FROM utenti
WHERE id NOT IN (
    SELECT DISTINCT utente_id
    FROM visualizzazioni
    WHERE utente_id IS NOT NULL
);
```

Query Tipo 4: Ricerca con Date

Pattern: "Eventi in un periodo" o "Dati dell'anno corrente"

```sql
-- Intervallo di date
WHERE data_campo BETWEEN '2024-01-01' AND '2024-12-31'

-- Anno corrente
WHERE YEAR(data_campo) = YEAR(CURDATE())

-- Ultimo mese
WHERE data_campo >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
```

---

6. PROGETTO WEB - SCHEMA PRATICO

Struttura Minima Sufficiente

```
1. Homepage
   - Descrizione servizio
   - Login/Registrazione

2. Area Utente
   - Dashboard personale
   - CRUD principale (inserimento/modifica)
   - Visualizzazione dati
```

```
3. Area Admin (se richiesta)
   - Gestione utenti
   - Statistiche/report
```

Tecnologie Standard da Citare

```
Frontend: HTML5 + CSS3 + JavaScript
Backend: PHP + MySQL
Server: Apache/Nginx
Pattern: MVC (Model-View-Controller)
Sicurezza: HTTPS + prepared statements
```

---

7. CODIFICA - TEMPLATE UNIVERSALE

Struttura PHP Standard

```php
<?php
session_start();

// 1. CONNESSIONE DATABASE
$host = "localhost";
$username = "root";
$password = "";
$database = "nome_db";

$conn = new mysqli($host, $username, $password, $database);
if ($conn->connect_error) {
    die("Errore connessione: " . $conn->connect_error);
}

// 2. PROCESSING FORM (se presente)
if ($_POST) {
    $campo1 = $conn->real_escape_string($_POST['campo1']);

    $sql = "INSERT INTO tabella (campo1) VALUES (?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $campo1);
    $stmt->execute();
}

// 3. QUERY PRINCIPALE
$sql = "SELECT * FROM tabella WHERE condizione ORDER BY campo";
```

```php
$result = $conn->query($sql);
?>

<!DOCTYPE html>
<html>
<head>
    <title>Titolo Pagina</title>
    <meta charset="UTF-8">
</head>
<body>
    <h1>Intestazione</h1>

    <!-- 4. VISUALIZZAZIONE DATI -->
    <table border="1">
        <tr><th>Campo1</th><th>Campo2</th></tr>
        <?php while($row = $result->fetch_assoc()): ?>
        <tr>
            <td><?= htmlspecialchars($row['campo1']) ?></td>
            <td><?= htmlspecialchars($row['campo2']) ?></td>
        </tr>
        <?php endwhile; ?>
    </table>

    <!-- 5. FORM INPUT (se richiesto) -->
    <form method="POST">
        <input type="text" name="campo1" placeholder="Inserisci campo1"
required>
        <input type="submit" value="Salva">
    </form>
</body>
</html>
```

---

8. TRUCCHI OPERATIVI

Per Recuperare Tempo

1. **Schema E/R incompleto?** → Aggiungere solo entità principali + 1-2 relazioni
2. **Query complesse?** → Scrivere query parziale + commento con logica
3. **Codice non funziona?** → Lasciare commenti che spiegano cosa dovrebbe fare

Per Massimizzare Punteggio

1. **Prima le query semplici** (elenchi e JOIN)
2. **Schema logico sempre completo** (più importante del codice)
3. **Seconda parte:** scegliere domande su argomenti certi

Gestione Errori Comuni

Errore: Query senza risultati **Fix:** Controllare nomi tabelle e campi, verificare JOIN

Errore: Syntax error SQL
Fix: Controllare virgole, apici, parentesi

Errore: Cardinalità sbagliate **Fix:** Applicare regola "chi può averne molti"

Questa guida ti porta dal foglio bianco al compito completato seguendo procedure meccaniche e ripetibili.