Università di Catania

# PROGETTO BASI DI DATI

Gestione delle informazioni relative a un Gioco di Ruolo

Sciacca Santo Alessandro

1000035385

A.A. 2023/2024

# INDICE

# Introduzione

Verrà sviluppato un database al fine di gestire una sessione in un gioco di ruolo.
Ogni partecipante potrà scegliere cosa fare nella storia, che arma usare e che abilità lanciare.
Colui che gestirà il database sarà il Controllore stesso dei nemici e deciderà in che modo essi si approccino alla storia.

# Progettazione Concettuale

## Analisi dei requisiti

Il database gestirà le varie informazioni di una singola sessione di un gioco di ruolo.

Ogni giocatore avrà la possibilità di assegnare un nome al proprio personaggio, e gestirgli l'equipaggiamento.

Ogni Personaggio avrà un id univoco, un nome, l'id del giocatore che lo controlla, un livello, Punti Vita, Mana, una Arma e diverse Abilità.

Ogni Arma avrà diverse caratteristiche, come l'id, il nome, la tipologia e il danno.

Ogni Abilità avrà un id, il nome, la tipologia, il danno e la relativa descrizione.

Giocatore invece avrà le informazioni inerenti ai giocatori della sessione, con relativi dati anagrafici e id unificativi.

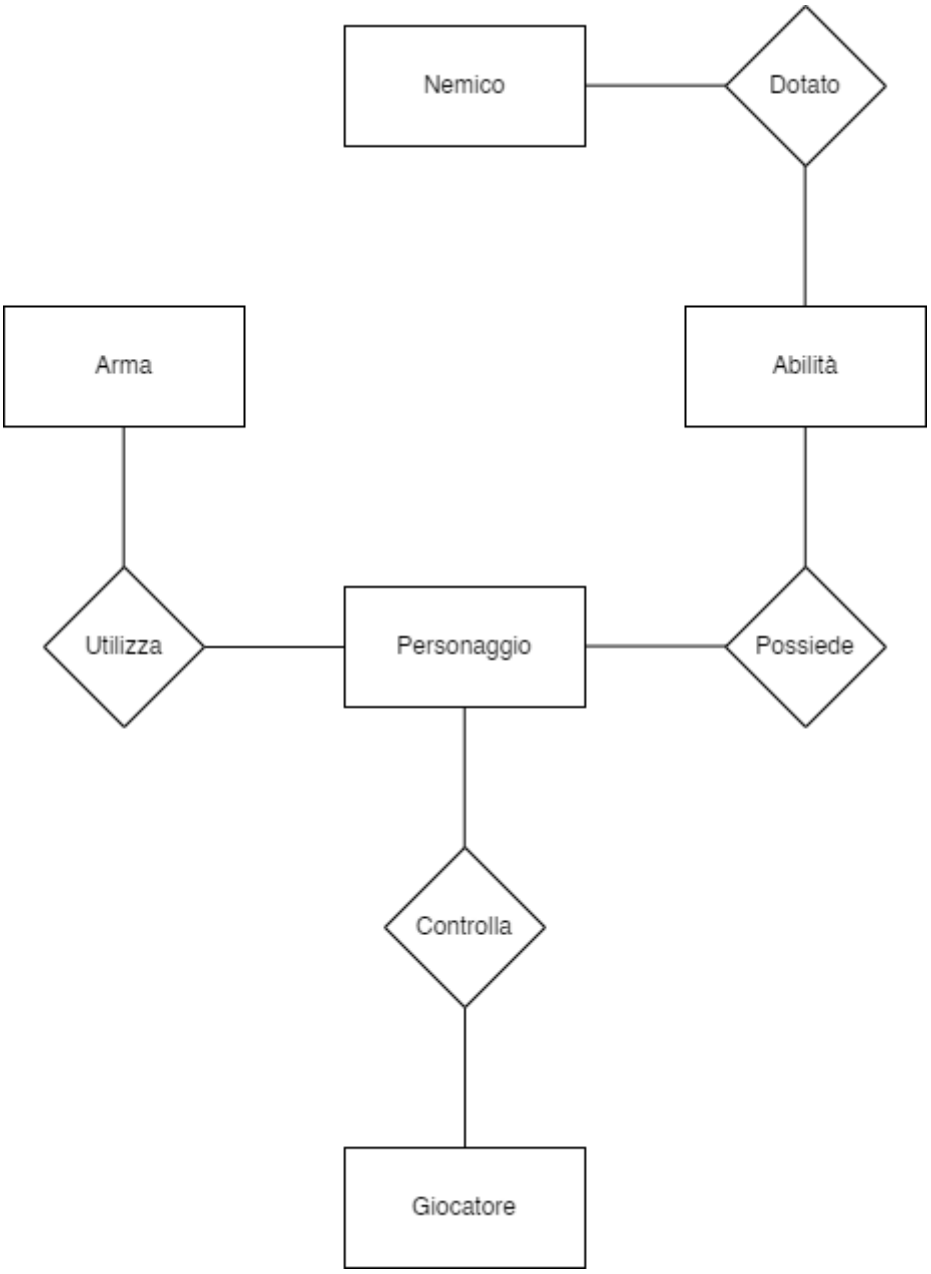
Avremo anche dei Nemici aventi id, nome, vita, e numerose abilita.

Tipologia sarà un valore che specifica se l'arma o abilità in questione sia da corta, media o lunga distanza.

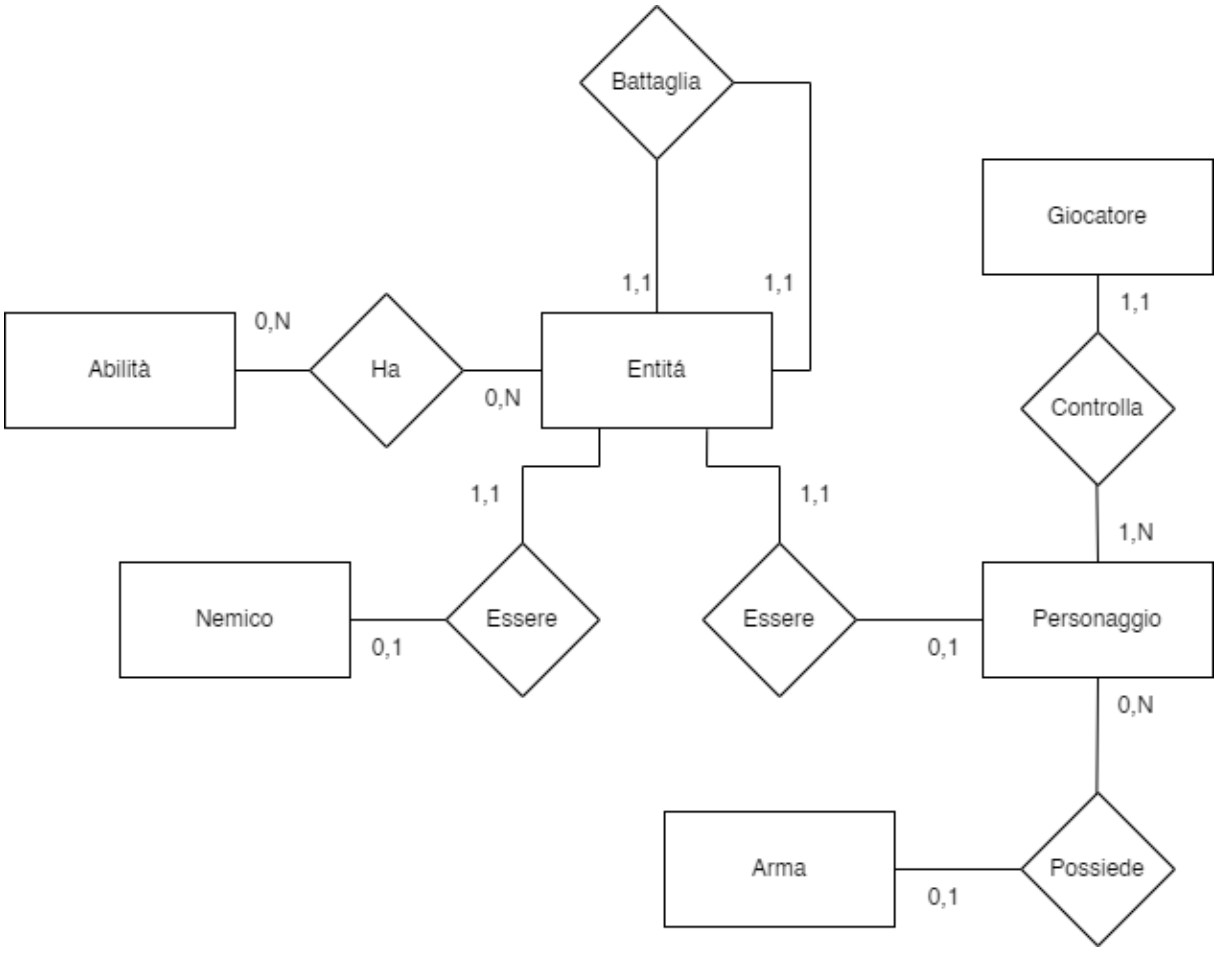
## Glossario dei termini

| Termine | Descrizione | Sinonimi | Collegamenti |
|---|---|---|---|
| Giocatore | Persona giocante che controlla uno o più personaggi | | Personaggio |
| Personaggio | Entità controllata dal giocatore al fine di eseguire azioni all'interno dell'avventura | | Giocatore, Armi, Abilitá |
| Arma | È un oggetto fine al far danno ad un'altra entità, che sia personaggio o nemico. | | Personaggio, Nemico, Tipologia |
| Abilità | Una azione particolare, effettuabile da un personaggio o nemico | | Personaggio, Nemico, Tipologia |
| Tipologia | Valore inerente alla distanza minima a cui una arma o abilità possa colpire | distanza | Arma, Abilità |
| Nemico | Entità nemiche aventi l'obiettivo di rallentare o eventualmente uccidere i giocatori | | Arma, Abilità |

## Schema Scheletro



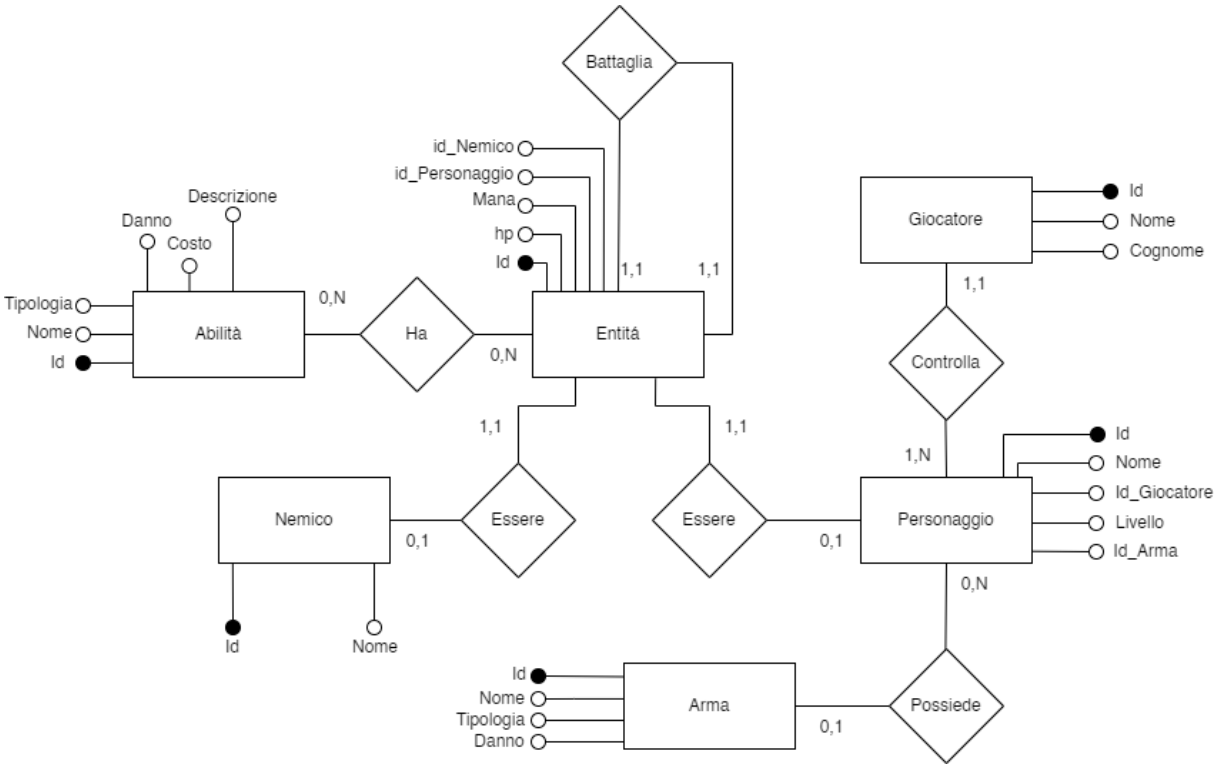
## Schema Intermedio




È stato ritenuto opportuno aggiungere una entità che colleghi Personaggio e Nemico, al fine di rendere il collegamento con Abilità più comprensibile.

È stata aggiunta anche la relazione battaglia, che permetterà a una entità qualsiasi, di poter interagire con una qualsiasi altra entità.

## Schema Finale




Non sono presenti ridondanze, da andare ad analizzare.

## Vincoli non esprimibili dal diagramma ER

Una Entità non può essere contemporaneamente sia un Nemico che un Personaggio.

Una Entità, Personaggio o Nemico che sia, può combattere con qualsiasi altra entità, compreso sé stessa.

La relazione battaglia ha obiettivo quello di ospitare al suo interno l'id di due Entità, e un id di un'abilità, andando quindi a intendere che la prima esegui tale abilità nei confronti della seconda.

l'Abilità con id uguale a zero è un attacco con arma, azione possibile solamente da una entità, non nemica, che la possiede.

## Dizionario dei dati

| Entità | Descrizione | Attributi | Identificatore |
|---|---|---|---|
| Giocatore | Persona che controlla un personaggio all'interno delle varie sessioni di gioco. | id, Nome, Cognome | id |
| Personaggio | Personaggio controllato dal giocatore | Id, Nome, Id_Giocatore, Livello, Id_Arma | id |
| Arma | Espediente fine al far danno ad altre entità | Id, Nome, Tipologia, Danno | id |
| Nemico | Entità ostile ai personaggi, ha come fine unico quello di eliminarli | Id, Nome | id |
| Abilità | Insieme di magie fini al fare danno, utilizzabili sia da Nemici che da Personaggi, hanno un costo che va sottratto ogni qualvolta ne venga usata una | Id, Nome, Tipologia, Danno, Costo, Descrizione | id |
| Entità | Identificativo di un essere presente nella storia giocata, Nemico o Personaggio che sia | Id, hp, Mana, Id_Personaggio, Id_Nemico | id |

## Dizionario delle Relazioni

| Associazione | Entità Partecipanti | Descrizione | Attributi |
|---|---|---|---|
| Controlla | Giocatore, Personaggio | Un giocatore controlla uno o più personaggi. | |
| Possiede | Personaggio, Arma | Un personaggio può possedere al massimo una singola arma | |
| Essere | Nemico, Personaggio, Entità | Una entità può essere un Personaggio o un Nemico | |
| Ha | Entità, Abilità | Una entità ha a disposizione diverse abilità. | |
| Battaglia | Entità | Una entità può combattere con un'altra Entità (o sé stessa) | |

# Progettazione Logica

## Stime

Le seguenti stime sono riferite alla sessione di gioco a cui il database è correlato, ciò implica che le abilità presenti e armi, possano essere aggiunte o rimosse sulla base delle richieste della storia narrata, e che i giocatori siano strettamente correlati ai personaggi giocanti.

I giocatori possono controllare più di un singolo personaggio, ma al minimo uno per essere presenti, date queste informazioni possiamo stimare che ne siano presenti 3, di cui uno controllante 2 personaggi.

Come detto prima, i personaggi possono essere al minimo 1; quindi, supponiamo di trattare 10 personaggi di cui 6 morti (0 hp), e in media una nuova aggiunta ogni settimana (implica che un personaggio in media ogni settimana muoia e se ne venga creato uno nuovo).

Supponiamo di avere anche 12 armi con una nuova aggiunta in media ogni settimana.

Discorso analogo per le abilità, supponiamo di averne dieci con una nuova aggiunta ogni mese.

Supponiamo anche che nella nostra storia narrata i nemici da affrontare siano nove in totale.

## Tabella dei volumi

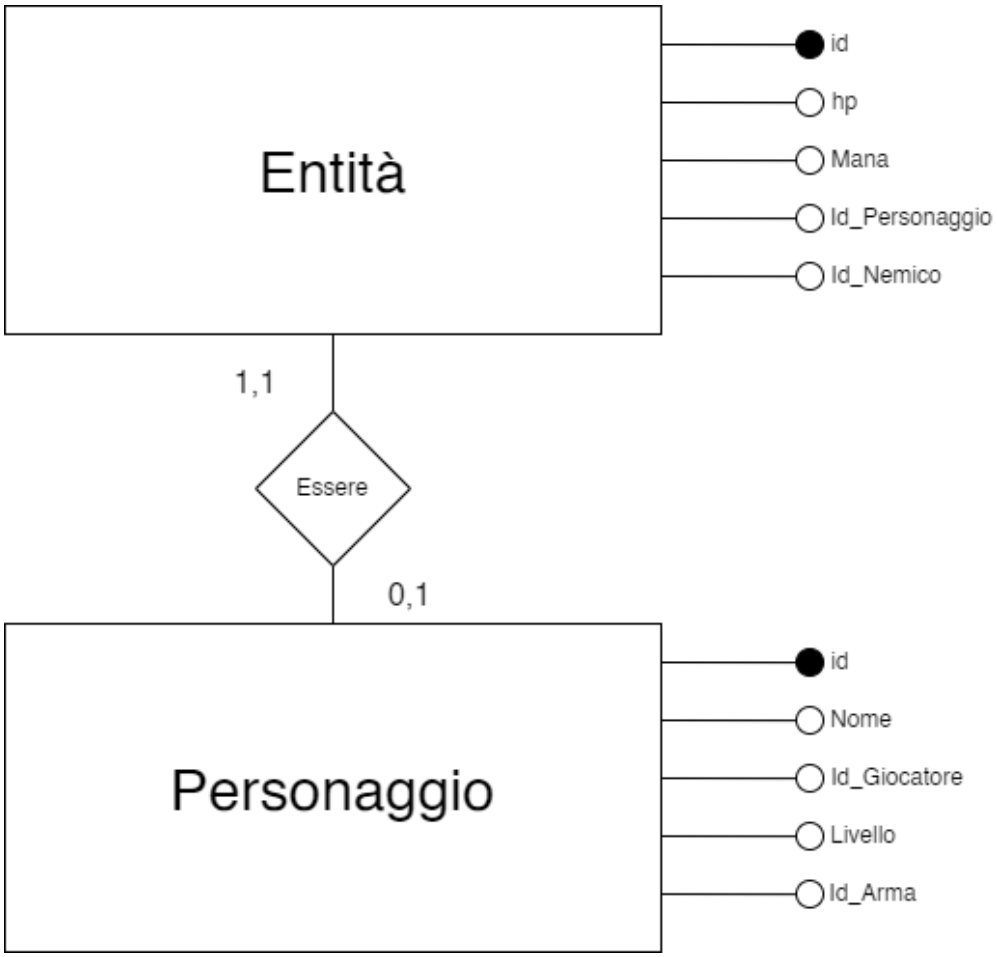
| Concetto | Tipo | Volume |
|---|---|---|
| Giocatore | E | 3 |
| Personaggio | E | 10 |
| Arma | E | 12 |
| Nemico | E | 9 |
| Abilità | E | 10 |

## Tabella delle frequenze

| Operazione | Descrizione | Frequenza | Tipo |
|---|---|---|---|
| O1 | Aggiunta nuovo personaggio | 1/settimana | Interattiva |
| O2 | Aggiunta nuova abilità | 1/mese | Interattiva |
| O3 | Aggiunta nuova arma | 1/settimana | Interattiva |
| O4 | Visualizzazione dell'arma posseduta da più Personaggi | 1/settimana | batch |
| O5 | Visualizzazione della abilità col danno maggiore | 1/mese | batch |
| O6 | Visualizzazione del livello medio dei Personaggi | 1/settimana | batch |

## Schemi delle operazioni

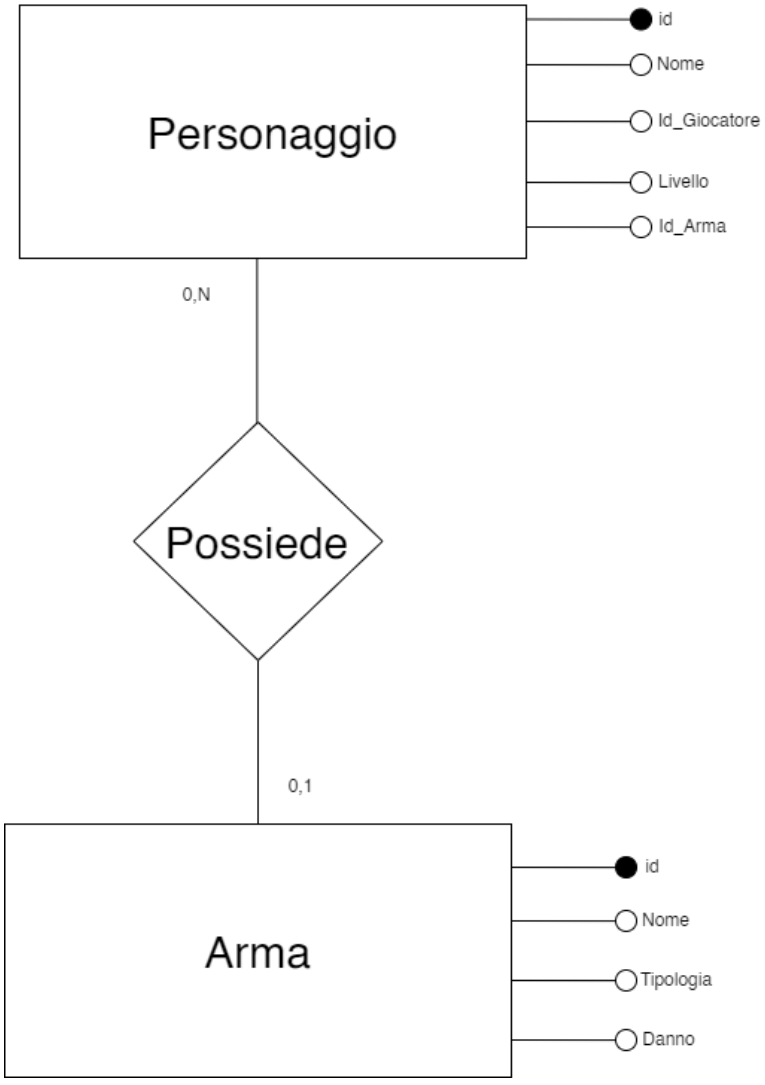
O1. Aggiunta di un nuovo personaggio




O2. Aggiunta di una nuova Abilità

O3. Aggiunta di una nuova Arma


O4. Visualizzare l'arma posseduta da più Personaggi



O5. Visualizzare l’abilità con il danno maggiore


O6. Visualizzare la media aritmetica dei livelli dei Personaggi

## Schema Logico

Entità (id, hp, mana, id_Personaggio, id_Nemico);

Nemico (id, nome);

Personaggio (id, nome, id_Giocatore, livello, id_Arma);

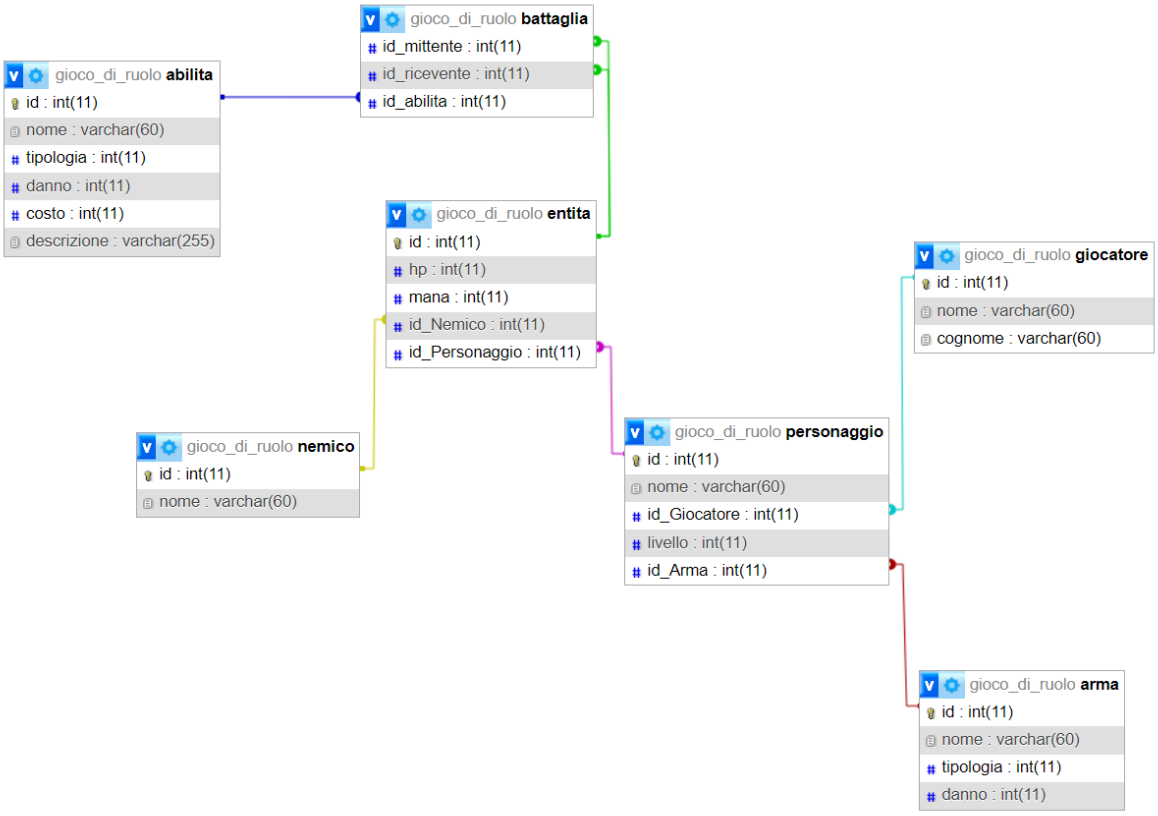
Giocatore (id, nome, cognome);

Arma (id, nome, tipologia, danno);

Abilità (id, nome, tipologia, danno, costo, descrizione);

Battaglia (id_Mittente, id_Ricevente, id_Abilità)

## Schema Fisico



# Progettazione Fisica

Esempi di Create utilizzate nel database:

```
CREATE TABLE Entita(
    id INT NOT NULL,
    hp int,
    mana int,
    id_Nemico INT,
    id_Personaggio INT,
    PRIMARY KEY (id),
    FOREIGN KEY (id_Nemico) REFERENCES  Nemico(id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
    FOREIGN KEY (id_Personaggio) REFERENCES  Personaggio(id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
    CHECK (id_Nemico IS NOT NULL XOR id_Personaggio IS NOT NULL)
);
```

```
CREATE TABLE Battaglia (
    id_mittente INT NOT NULL,
    id_ricevente INT NOT NULL,
    id_abilita INT NOT NULL,
    FOREIGN KEY (id_mittente) REFERENCES Entita(id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
    FOREIGN KEY (id_ricevente) REFERENCES Entita(id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
    FOREIGN KEY (id_abilita) REFERENCES Abilita(id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE
);
```

```
CREATE TABLE Abilita(
    id INT NOT NULL,
    nome varchar(60),
    tipologia INT,
    danno INT,
    costo INT,
    descrizione varchar(255),
    PRIMARY KEY (id)
);
```

È possibile approfondire le Create [Premendo Qui]

# Creazione dei Trigger

1) Abbiamo bisogno di un Trigger che prima di una nuova insert all'interno di Battaglia, vada a verificare e in caso modificare tutti i dati relativi delle entità prese in considerazione.

```
1) DELIMITER //
2)
3) CREATE TRIGGER before_insert_battaglia
4) BEFORE INSERT ON Battaglia
5) FOR EACH ROW
6) BEGIN
7)     DECLARE attacker_hp INT;
8)     DECLARE attacker_mana INT;
9)     DECLARE defender_hp INT;
10)     DECLARE defender_mana INT;
11)     DECLARE attacker_damage INT;
12)     DECLARE ability_cost INT;
13)     DECLARE attacker_is_character BOOL;
14)
15)     -- Ottieni le informazioni sull'attaccante e sul difensore
16)     SELECT hp, mana, id_Personaggio IS NOT NULL INTO attacker_hp,
   attacker_mana, attacker_is_character
17)     FROM Entita WHERE id = NEW.id_mittente;
18)
19)     SELECT hp, mana INTO defender_hp, defender_mana
20)     FROM Entita WHERE id = NEW.id_ricevente;
21)
22)     -- Controlla se entrambi gli enti sono vivi
23)     IF attacker_hp > 0 AND defender_hp > 0 THEN
24)         -- Se l'abilità è Attacco con arma (id_abilita = 0)
25)         IF NEW.id_abilita = 0 THEN
26)             -- Verifica se l'attaccante è un personaggio o un nemico
27)             IF attacker_is_character THEN
28)                 -- Ottieni il danno dell'arma del personaggio
29)                 SELECT Arma.danno INTO attacker_damage FROM Entita,
   Personaggio, Arma
30)                 WHERE Entita.id = NEW.id_mittente and
   Entita.id_Personaggio = Personaggio.id and Personaggio.id_Arma =
   Arma.id;
31)
32)                 -- Infliggi il danno al difensore
33)                 IF NEW.id_mittente = NEW.id_ricevente THEN
34)                     SET attacker_hp = GREATEST(attacker_hp -
   attacker_damage, 0);
35)                 ELSE
36)                     SET defender_hp = GREATEST(defender_hp -
   attacker_damage, 0);
```

```
37)                 END IF;
38)             ELSE
39)                 -- L'attaccante non può essere un nemico, generare
    errore
40)                 SIGNAL SQLSTATE '45000'
41)                 SET MESSAGE_TEXT = 'Impossibile attaccare con arma: l
    attaccante non può essere un nemico.';
42)             END IF;
43)         ELSE
44)             -- L'abilità non è Attacco con arma
45)             -- Ottieni il costo e il danno dell'abilità
46)             SELECT costo, danno INTO ability_cost, attacker_damage
47)             FROM Abilita
48)             WHERE id = NEW.id_abilita;
49)
50)             -- Verifica se l'attaccante ha abbastanza mana per
    utilizzare l'abilità
51)             IF attacker_mana < ability_cost THEN
52)                 SIGNAL SQLSTATE '45000'
53)                 SET MESSAGE_TEXT = 'Impossibile avviare la battaglia:
    mana insufficiente per utilizzare l abilità.';
54)             ELSE
55)                 -- Infliggi il danno al difensore e riduci il mana
    dell'attaccante
56)                 IF NEW.id_mittente = NEW.id_ricevente THEN
57)                     SET attacker_hp = GREATEST(attacker_hp -
    attacker_damage, 0);
58)                 ELSE
59)                     SET defender_hp = GREATEST(defender_hp -
    attacker_damage, 0);
60)                 END IF;
61)                 SET attacker_mana = attacker_mana - ability_cost;
62)             END IF;
63)         END IF;
64)
65)         -- Aggiorna le informazioni sull'attaccante e sul difensore
66)         UPDATE Entita SET hp = defender_hp WHERE id =
    NEW.id_ricevente;
67)         UPDATE Entita SET hp = attacker_hp, mana = attacker_mana
    WHERE id = NEW.id_mittente;
68)     ELSE
69)         -- Uno o entrambi gli enti non sono vivi, genera errore
70)         SIGNAL SQLSTATE '45000'
71)         SET MESSAGE_TEXT = 'Impossibile avviare la battaglia: uno o
    entrambi gli enti non sono vivi.';
72)     END IF;
73) END;
74) //
75) DELIMITER ;
```

2) Abbiamo bisogno di un trigger che gestisca i livelli dei Personaggio, ogni update controlla se sono vivi, se il nuovo livello è maggiore del precedente, e se i controlli passano, va ad aggiungere 10hp e 5 mana alla entità associata a quel personaggio.

```
1) DELIMITER //
2)
3) CREATE TRIGGER before_update_personaggio
4) BEFORE UPDATE ON Personaggio
5) FOR EACH ROW
6) BEGIN
7)     DECLARE livello_precedente INT;
8)     DECLARE livello_successivo INT;
9)     DECLARE differenza_livello INT;
10)     DECLARE hp_incremento INT;
11)     DECLARE mana_incremento INT;
12)
13)     -- Ottieni il livello precedente e il livello successivo
14)     SELECT OLD.livello, NEW.livello INTO livello_precedente,
   livello_successivo;
15)
16)     -- Controlla che il personaggio sia vivo
17)     IF (SELECT hp FROM Entita WHERE id_Personaggio = NEW.id) <= 0
   THEN
18)         SIGNAL SQLSTATE '45000'
19)         SET MESSAGE_TEXT = 'Impossibile aggiornare il livello: il
   personaggio non è vivo.';
20)     END IF;
21)
22)     -- Verifica se il nuovo livello è maggiore del precedente
23)     IF livello_successivo > livello_precedente THEN
24)         -- Calcola la differenza di livello
25)         SET differenza_livello = livello_successivo -
   livello_precedente;
26)
27)         -- Calcola l'incremento di HP e mana per ogni livello
   aumentato
28)         SET hp_incremento = differenza_livello * 10;
29)         SET mana_incremento = differenza_livello * 5;
30)
31)         -- Aggiorna le colonne hp e mana dell'entità associata al
   personaggio
32)         UPDATE Entita
33)         SET hp = hp + hp_incremento,
34)             mana = mana + mana_incremento
35)         WHERE id_Personaggio = NEW.id;
36)     ELSE
37)         -- Se il nuovo livello non è maggiore del precedente, genera
   un errore
38)         SIGNAL SQLSTATE '45000'
```

```
39)         SET MESSAGE_TEXT = 'Impossibile aggiornare il livello: il
   nuovo livello deve essere superiore al precedente.';
40)     END IF;
41) END;
42) //
43) DELIMITER ;
```

Entrambi i trigger possono essere visualizzati [Premendo qui]

# Operazioni

O1 Aggiunta nuovo **Personaggio**

```
INSERT INTO Personaggio VALUES
(10, "Darius", 0, 5, 5);

INSERT INTO Entita (id, hp, mana, id_Personaggio) VALUES
(19, 50, 15 ,10);
```

O2 Aggiunta nuova **Abilità**

```
INSERT INTO Abilita VALUES
(10,"Pugno infiammato", 2, 10, 20,"Sferra un pugno infiammato");
```

O3 Aggiunta nuova **Arma**

```
INSERT INTO Arma VALUES
(12, "Spada curva", 2, 13);
```

O4 Visualizzazione dell'**Arma** posseduta da più **Personaggio**

```
SELECT arma.nome AS Nome_Arma, COUNT(*) AS Numero_Personaggi_Che_la_usano
FROM personaggio, arma
WHERE personaggio.id_Arma=arma.id
GROUP BY personaggio.id_Arma
ORDER BY COUNT(*) DESC
LIMIT 1;
```

O5 Visualizzazione della **Abilità** col danno maggiore

```
SELECT abilita.nome as Abilità, abilita.danno
FROM abilita
ORDER BY abilita.danno DESC
LIMIT 1
```

O6 Visualizzazione del livello medio dei **Personaggio**

```
SELECT AVG(personaggio.livello) AS LivelloMedio
FROM personaggio
```