BLADES & MAGIC

ELABORATO PER IL CORSO DI BASI DI DATI

Lorenzo Simoncini matr. 0000765820

Pasquale Nardella matr. 0000757949

Matteo Tallevi Diotallevi matr. 0000765758

STRUTTURA DELL'ELABORATO

La presente documentazione tratta nel dettaglio la progettazione e l'implementazione dell'elaborato "Blades & Magic" di Lorenzo Simoncini, Pasquale Nardella e Matteo Tallevi Diotallevi, ed è strutturata come segue:

1 – INTRODUZIONE

Il progetto consiste nella realizzazione di un sistema database che funga da supporto a un server di gioco per il videogame "Blades & Magic", un MMORPG (massive multiplayer online role-playing game) ovvero un gioco di ruolo online dove i giocatori possono affrontare missioni sia cooperando tra loro che da soli, oppure sfidarsi in duelli. Il database dovrà non solo immagazzinare informazioni riguardo ai giocatori iscritti, alle partite, alle squadre e alle missioni effettuate, ma dovrà anche modellare aspetti relativi al mondo di gioco, come ad esempio i premi ottenibili da ciascuna missione, così come l'inventario di ogni giocatore, i luoghi da visitare e le attività offerte da ogni luogo. Si adotta questa soluzione, anziché due sistemi informativi separati perché i concetti modellati e le informazioni immagazzinate sono strettamente collegate tra loro, e si preferisce avere un accesso più uniforme ai dati. Questo comporta che ci saranno interfacce di accesso al database diverse, a seconda che l'utente sia un giocatore o l'amministratore di sistema. Ad esempio, si potrebbe pensare che il giocatore debba poter interrogare il database per ottenere i suoi amici attualmente connessi online, le missioni che stanno svolgendo, eventuali tornei a cui partecipare, ma anche ottenere informazioni sui posti del mondo da visitare, o delle missioni che deve svolgere per poter vincere un oggetto che gli interessa. Dall'altro lato, l'amministratore di sistema deve poter agire sul database per poter modificare i contenuti del gioco, ad esempio aggiungendo nuove missioni, nuovi personaggi giocabili, (aggiornamenti al gioco) ma anche creando tornei, sfide ed eventi a tempo a cui i giocatori possano partecipare, per mantenere l'esperienza di gioco sempre fresca e aggiornata.

Il Sistema informativo offrirà quindi due client distinti. Il client del giocatore sarà direttamente integrato nel motore del videogioco stesso, rilasciato pubblicamente, mentre il client dell'amministratore di sistema sarà una applicazione specifica che permetta l'accesso diretto al database dai soli membri del gruppo di sviluppo, e solo loro ne possederanno una copia. In questa relazione verrà esposta un'interfaccia client d'amministrazione, che implementi sia le funzioni che verranno usate dal giocatore (per motivi di completezza e test) che le funzioni degli amministratori.

2 – ANALISI DEI REQUISITI

2.1 - Requisiti in linguaggio naturale

La seguente descrizione riporta in linguaggio naturale i requisiti per il nostro sistema informativo:

"La software house WareSoft© richiede un sistema informatico di supporto alla realizzazione di un gioco di ruolo online di tipo MMORPG, per la gestione delle informazioni relative ai giocatori e al mondo di gioco. Il database di informazioni verrà acceduto sia dai giocatori, tramite il videogame stesso, sia dagli sviluppatori del gioco e dai curatori, che durante le manutenzioni dei server devono poter accedere ai dati del sistema informativo per poter aggiornare le informazioni. Il sistema deve poter memorizzare la lista degli utenti iscritti, ciascuno di essi avrà un proprio account al quale sarà associato un indirizzo email e password. Gli account sono gratuiti, ma è possibile acquistare un account premium per ottenere bonus all'interno del gioco. Il giocatore ha la possibilità, prima di iniziare una nuova avventura, di creare il suo personaggio, scegliendone la razza. Ad ogni account possono essere associati fino a 4 personaggi. La creazione di un personaggio richiede la scelta di una classe: questa scelta è importante perché ogni classe determina le statistiche del personaggio e le mosse che esso potrà imparare durante il gioco (insegnate da appositi NPC). Una volta che il gioco inizia, il giocatore può subito iniziare a esplorare il mondo virtuale per completare missioni e ottenere ricompense. I nuovi giocatori partono tutti dalla stessa zona. Il mondo di gioco è diviso in macro-regioni. Ogni regione è composta da più zone che contengono luoghi di interesse. Un luogo di interesse può essere uno dei seguenti tipi: un villaggio (abitato da NPC ciascuno dei quali ha un ruolo diverso), un dungeon (dove è possibile affrontare mostri) e un'arena (luogo dove si tengono tornei PVP tra i giocatori). I villaggi sono abitati da molti tipi di NPC, ciascuno dei quali ha un ruolo diverso. Ci sono i commercianti (offrono oggetti in cambio di denaro), gli allenatori (possono insegnare abilità), i crafter (costruiscono oggetti a partire dalle materie prime) e infine i comuni abitanti del villaggio (potrebbero proporre delle quest). Le quest sono missioni che possono essere completate per ottenere ricompense dall'NPC che le ha proposte. Esse consistono in eliminare un certo numero di nemici, oppure collezionare un certo numero di oggetti, oppure sconfiggere il boss di un determinato dungeon. Le quest non possono ricadere in più di queste categorie. Le ricompense consistono in denaro e/o un oggetto raro. I personaggi controllati dal giocatore collezionano oggetti di varie categorie: equipaggiamenti (armi e armature che aumentano le statistiche), oggetti di quest (senza alcun effetto, ma necessari al completamento delle missioni), valute (usati per essere scambiati con oggetti), materiali (usati per costruire oggetti più complessi) e infine consumabili (pozioni da usare in combattimento). Gli oggetti che possiede ogni giocatore sono conservati nel suo inventario, che ha 80 slots. Ogni slot può contenere più copie di una tipologia di oggetto. Il giocatore ha la possibilità di fondare un party o entrare in uno già esistente. Far parte di un party permette di affrontare, da soli o assieme ai propri amici (massimo 5) i vari dungeon. I dungeon sono percorsi popolati da molti nemici, e terminano con un nemico più forte, il boss. Ogni tipo di nemico, se sconfitto, può lasciare a terra (randomicamente) oggetti che vengono dati ad ogni membro del party. Nei duelli arena, invece, si scontrano tra loro due giocatori. Il vincitore riceve denaro ed esperienza. Sia le partite nei dungeon che i duelli nelle arene sono storicizzati per tenere traccia dei giocatori che hanno partecipato. Anche le quest completate sono storicizzate. Alcune quest possono essere rigiocate più volte mentre altre no."

2.2 - Estrazione dei concetti fondamentali

Individuiamo adesso le parole e le espressioni chiave che ci consentiranno di realizzare uno schema significativo del progetto e di raffinarlo successivamente per ottenere lo schema definitivo. I termini di rilievo appaiono nel testo con una sottolineatura:

*"La software house WareSoft© richiede un sistema informatico di supporto alla realizzazione di un gioco di ruolo online di tipo MMORPG, per la gestione delle informazioni relative ai giocatori e al mondo di gioco. Il database di informazioni verrà acceduto sia dai giocatori, tramite il videogame stesso, sia dagli sviluppatori del gioco e dai curatori, che durante le manutenzioni dei server devono poter accedere ai dati del sistema informativo per poter aggiornare le informazioni. Il sistema deve poter memorizzare la lista degli utenti iscritti, ciascuno di essi avrà un proprio **account** al quale sarà associato un indirizzo email e password. Gli account sono gratuiti, ma è possibile acquistare un account premium per ottenere bonus all'interno del gioco. Il giocatore ha la possibilità, prima di iniziare una nuova avventura, di creare il suo **personaggio,** scegliendone la **razza**. Ad ogni account possono essere associati fino a 4 personaggi. La creazione di un personaggio richiede la scelta di una **classe**: questa scelta è importante perché ogni classe determina le statistiche del personaggio e le **mosse** che esso potrà imparare durante il gioco (insegnate da appositi NPC). Una volta che il gioco inizia, il giocatore può subito iniziare a esplorare il mondo virtuale per completare quest e ottenere ricompense. I nuovi giocatori partono tutti dalla stessa zona. Il mondo di gioco è diviso in **macro-regioni**. Ogni regione è composta da più **zone** che contengono **luoghi di interesse**. Un luogo di interesse può essere uno dei seguenti tipi: un **villaggio** (abitato da **NPC** ciascuno dei quali ha un ruolo diverso), un **dungeon** (dove è possibile affrontare **mostri**) e un'**arena** (luogo dove si tengono tornei PVP tra i giocatori). I villaggi sono abitati da molti tipi di NPC, ciascuno dei quali ha un ruolo diverso. Ci sono i **commercianti** (offrono **oggetti** in cambio di denaro), **gli allenatori** (possono insegnare abilità), i **crafter** (costruiscono oggetti a partire dalle materie prime) e infine i comuni **abitanti del villaggio** (potrebbero proporre delle quest). Le **quest** sono missioni che possono essere completate per ottenere **ricompense** dall'NPC che le ha proposte. Esse consistono in eliminare un certo numero di nemici, oppure collezionare un certo numero di oggetti, oppure sconfiggere il boss di un determinato dungeon. Le quest non possono ricadere in più di queste categorie. Le ricompense consistono in denaro e/o un oggetto raro. I personaggi controllati dal giocatore collezionano **oggetti** di varie categorie: **equipaggiamenti** (armi e armature che aumentano le statistiche), **oggetti di quest** (senza alcun effetto, ma necessari al completamento delle missioni), **valute** (usati per essere scambiati con oggetti), **materiali** (usati per costruire oggetti più complessi) e infine **consumabili** (pozioni da usare in combattimento). Gli oggetti che possiede ogni giocatore sono conservati nel suo **inventario**, che ha 80 slots. Ogni **slot** può contenere più copie di una tipologia di oggetto. Il giocatore ha la possibilità di fondare un **party** o entrare in uno già esistente. Far parte di un party permette di affrontare, da soli o assieme ai propri amici (massimo 5) i vari dungeon. I dungeon sono percorsi popolati da molti **nemici**, e terminano con un nemico più forte, il **boss**. Ogni tipo di nemico, se sconfitto, può lasciare a terra (randomicamente) oggetti che vengono dati ad ogni membro del party. Nei duelli arena, invece, si scontrano tra loro due giocatori. Il vincitore riceve denaro ed esperienza. Sia le partite nei dungeon che i duelli nelle arene sono storicizzati per tenere traccia dei giocatori che hanno partecipato. Anche le quest completate sono storicizzate. Alcune quest possono essere rigiocate più volte mentre altre no."*

La descrizione è già abbastanza chiara e completa. La software house che commissiona il sistema informativo ha ben chiaro quale sarà la struttura del database, perché sicuramente durante la fase di game design ha annotato con precisione i requisiti del sistema informativo richiesto. Occorre comunque fare alcune precisazioni.

- "Giocatore" non è un termine di rilievo perché nel database si tengono informazioni solo relative all'account. Nulla vieta ad una persona di creare due account diversi, fornendo due indirizzi email.

- Per "mostri" e "nemici" si intende lo stesso tipo di entità.

Lo schema del gioco completo risulterà molto vasto e poco leggibile da presentare in blocco, per cui per adesso si preferisce illustrarne una parte per volta. Le entità più importanti sono personaggio, oggetto, nemico e NPC, che sono quelle che hanno più interazioni tra tutte. Non è possibile presentare uno schema scheletro sintetico perché in esso verrebbero perse molte informazioni fondamentali.

3 - PROGETTO DELLO SCHEMA CONCETTUALE

Lo sviluppo dello schema Entity-Relationship procederà ora per fasi successive più o meno indipendenti da loro. Lo schema generale può essere suddiviso in più ambiti, in ciascuno dei quali si discuteranno soluzioni adottate, vincoli espressi e inespressi e altro. Gli schemi verranno poi raffinati e arricchiti. Nota: i termini utilizzati negli schemi, da questo momento in poi, verranno tradotti in lingua inglese.

- Account-Character
- Skill System and inventory
- Item Hierarchy (and relationships)
- World
- NPC
- Game

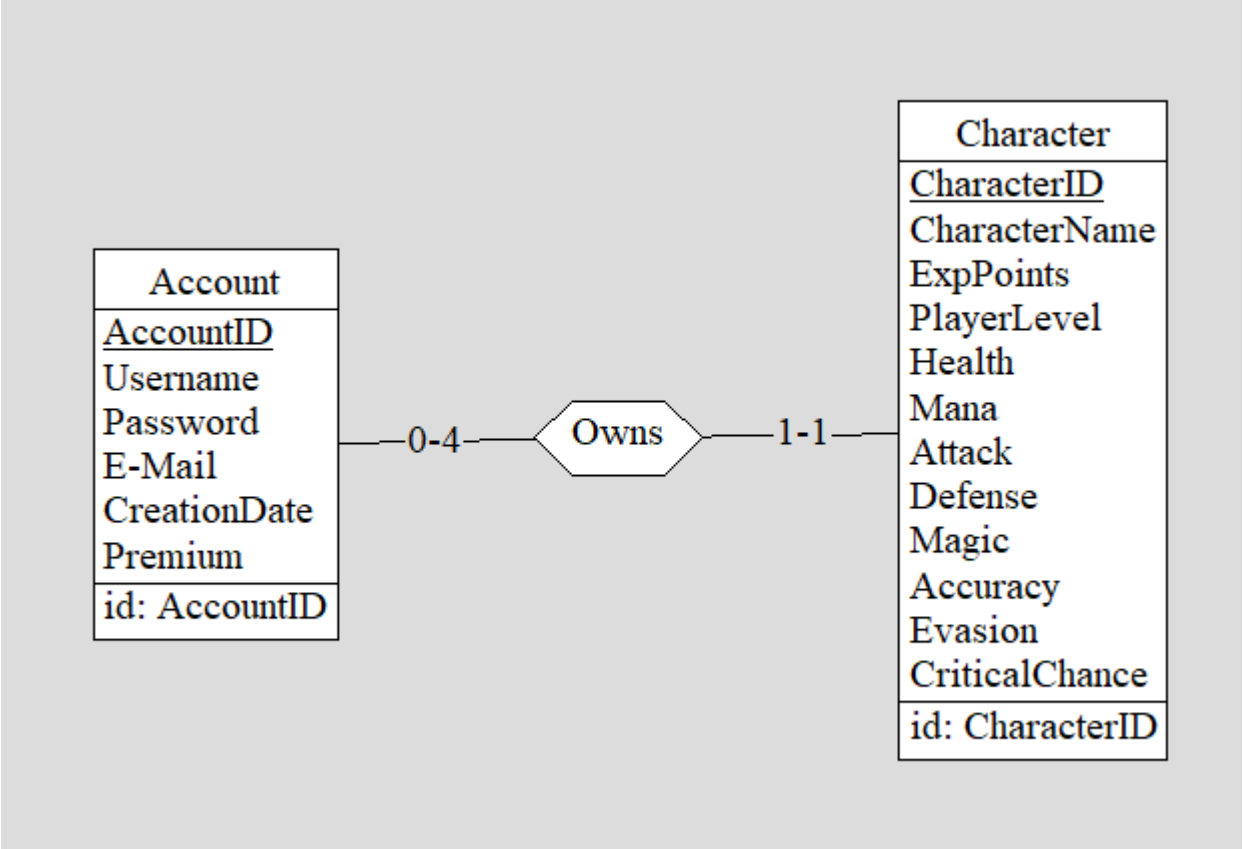
Lo schema concettuale nella sua versione finale si avvarrà delle seguenti entità e associazioni (per ciascuna è fornita una breve descrizione):

NOME	TIPO	DESCRIZIONE
ACCOUNT	E	Rappresenta l'account di un giocatore
ARENA	E	Rappresenta il luogo in cui avvengono le fight tra character
ARMOR	E	Rappresenta un tipo di equipment difensivo
AVAILABLE SKILL	R	Lega la class alle skill che può apprendere
BELONG	R	Lega il character alla sua race
CHARACTER	E	Rappresenta il personaggio di un giocatore
CLASS	E	Rappresenta la classe del personaggio
COMPLETE	R	Lega il dungeon alla quest che richiede di completarlo
COMPOSITION	R	Lega una region ai field di cui si compone
CONTAINMENT	R	Lega un field alle location che contiene
CONSUMABLE	E	Rappresenta un tipo di oggetto consumabile durante le battaglie
CRAFTER	E	Rappresenta un npc che crea oggetti a partire da altri
CRAFT LIST	R	Lega il crafter alle recipe che può creare
CURRENCY	E	Rappresenta un tipo di item scambiabile coi trader
DROP	R	Lega l'enemy agli item che può rilasciare
DUNGEON	E	Rappresenta un field ristretto popolato da enemy e da un boss
DUNGEON QUEST	E	Rappresenta una quest che chiede di completare un dungeon
ENEMY	E	Rappresenta un nemico da abbattere
ENEMY SKILL	R	Lega l'enemy alle skill che può usare
EQUIPMENT	E	Generalizzazione di armor, weapon
FETCH	R	Lega l'item alla fetch quest che richiede di raccoglierlo

FETCH QUEST	E	Rappresenta una quest che richieda la raccolta di un item
FIELD	E	Rappresenta una frazione della region
FIGHT	E	Rappresenta lo scontro tra due character
INGREDIENT	R	Lega la recipe agli item necessari per eseguirla
IS A	R	Lega il character alla classe a cui appartiene
ITEM	E	Generalizzazione di equipment, quest item, currency, material, consumable
KILL QUEST	E	Rappresenta una quest che richiede l'uccisione di enemy
LEARNED SKILL	R	Lega il character alle skill imparate
LOCATION	E	Generalizzazione di arena, village, dungeon
LOSER	R	Lega la fight al character perdente
LIVE	R	Lega l'npc al village in cui vive
MATERIAL	E	Rappresenta un item utile al crafting
MERCH	R	Lega il trader agli oggetti che vende
NPC	E	Generalizzazione di trader, crafter, villager, trainer
OWNS	R	Lega il character all'account a cui appartiene
PARTECIPATION	R	Lega il character al party a cui appartiene
PARTY	E	Rappresenta un insieme di giocatori riuniti per finire un dungeon
POPULATION_D	R	Lega l'enemy al dungeon in cui si trova
POPULATION_F	R	Lega l'enemy al field in cui si trova
PROPOSITION	R	Lega il villager alle quest che propone
QUEST	E	Generalizzazione di kill quest, fetch quest, dungeon quest
QUEST ITEM	E	Rappresenta un item necessario a completare una quest
QUEST LOG	R	Lega il character alle quest che ha completato
RACE	E	Rappresenta l'etnia del character
RECIPE	E	Rappresenta la ricetta per creare un item
REGION	E	Rappresenta la macro-zona del mondo di gioco
RESULT	R	Lega la recipe all'item che ne è risultato
RUN	R	Lega il party al dungeon che ha completato
SKILL	E	Rappresenta le abilità presenti nel mondo di gioco
STACK OF ITEM	R	Lega il character agli oggetti che possiede
TAKE PLACE	R	Lega la fight all'arena in cui si è svolta
TARGET	R	Lega la kill quest all'enemy che ne è bersaglio
TEACHING	R	Lega il trainer alle skill che può insegnare
TRADER	E	Rappresenta un npc che baratta item
TRAINER	E	Rappresenta un npc che insegna skill
VILLAGE	E	Rappresenta una location in cui vivono gli npc
VILLAGER	E	Rappresenta un npc che fornisce quest
WEAPON	E	Rappresenta un equipment offensivo
WINNER	R	Lega la fight al character vincente

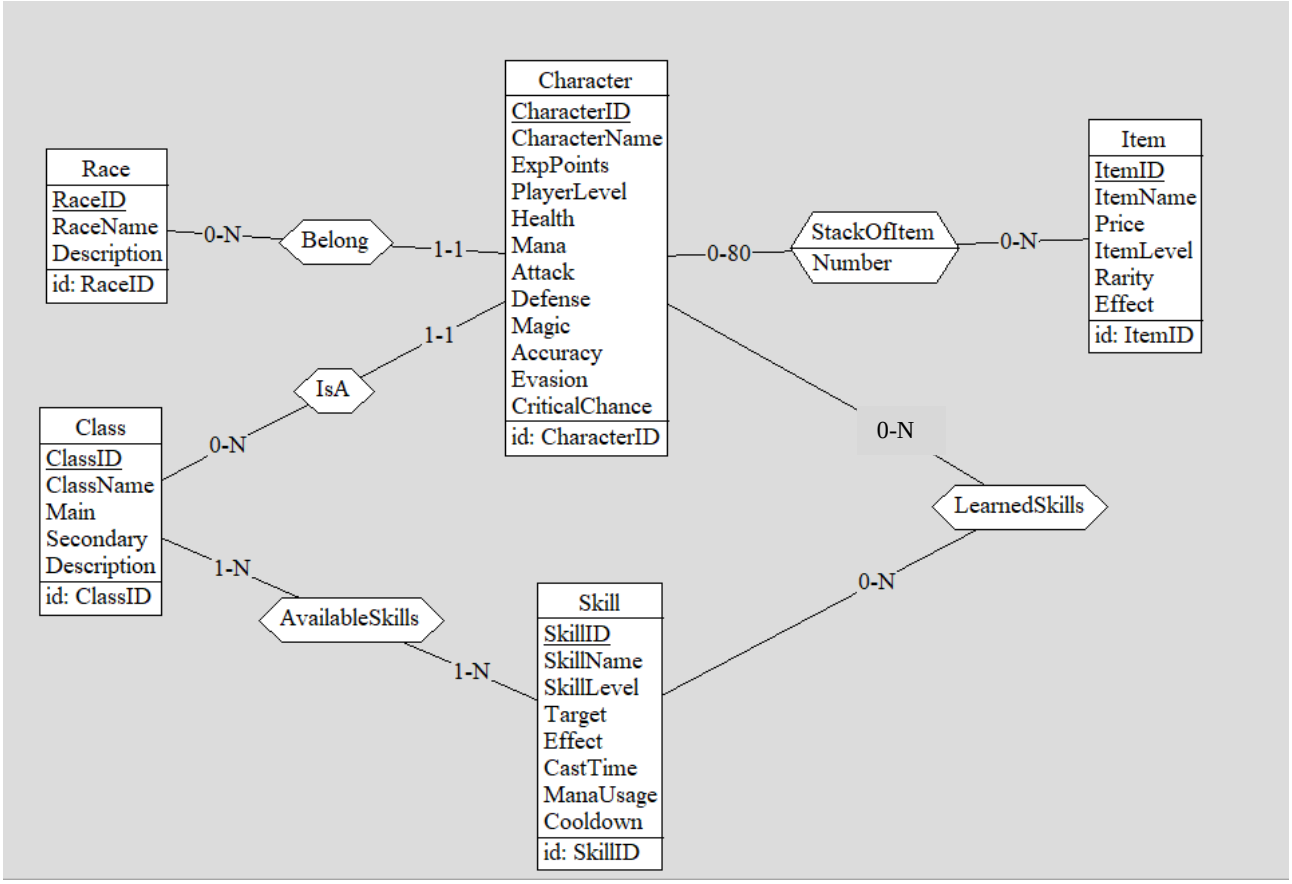
3.1 - Sviluppo dell'ambito "Account - Character"

Modellare questo aspetto del database è molto semplice. Sarà necessaria l'entità Account, legata all'entità Character per mezzo della relazione Own. Ad un account sono associati fino a 4 characters, ogni character è relativo ad un singolo account.



3.2 - Sviluppo dell'ambito "Skill system and inventory"

In quest'ambito si intende modellare la caratterizzazione dei personaggi che ogni giocatore può creare prima di iniziare a giocare. Saranno necessarie le entità Character, Class e Skill. Ogni personaggio appartiene ad una e una sola classe (mago, guerriero, tiratore ecc.), la classe determina le skill che possono essere apprese dal personaggio. Questo implica che ogni classe offre un possibile set di skill che si possono imparare, ed il personaggio durante il gioco ne imparerà un po' alla volta, tra esse. Inoltre nulla vieta che ci siano skill condivise tra più classi. Quando il giocatore vuole apprendere una nuova skill, il check relativo al fatto che la skill sia valida per la sua classe si può implementare in tanti modi, ad esempio verificando l'integrità del risultato fornito da due query. Per quanto riguarda l'inventario, si è detto che esso consiste in 80 slot oggetti, ciascuno dei quali può contenere più copie dello stesso oggetto. Per modellare questi aspetti serviranno la relazione StackOfItem e Item. Lo schema E-R risulterà come segue

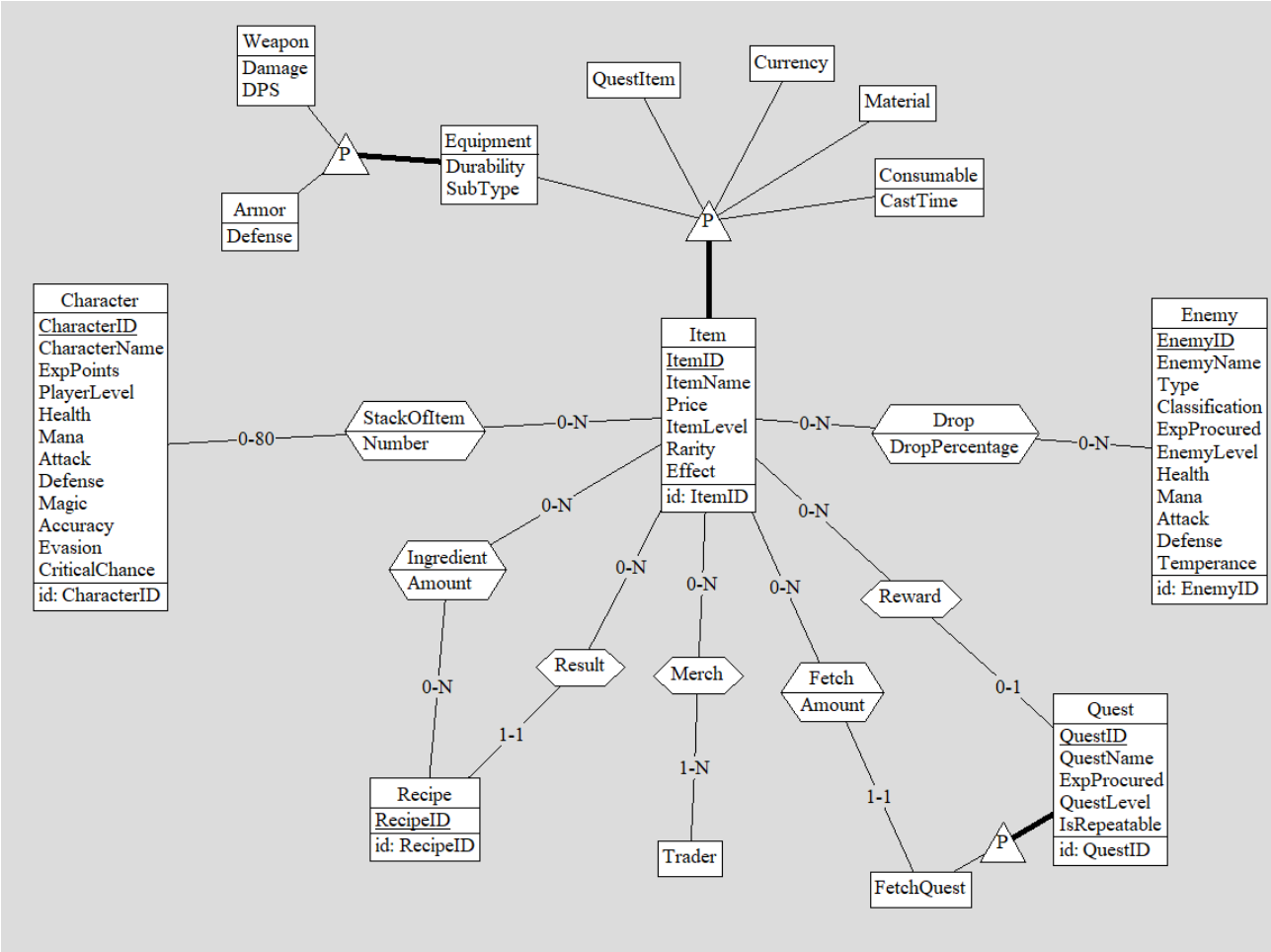


3.3 - Sviluppo dell'ambito "Item Hierarchy"

In quest'ambito si discuterà della gerarchia di oggetti che ogni personaggio colleziona. Le categorie sono le seguenti:

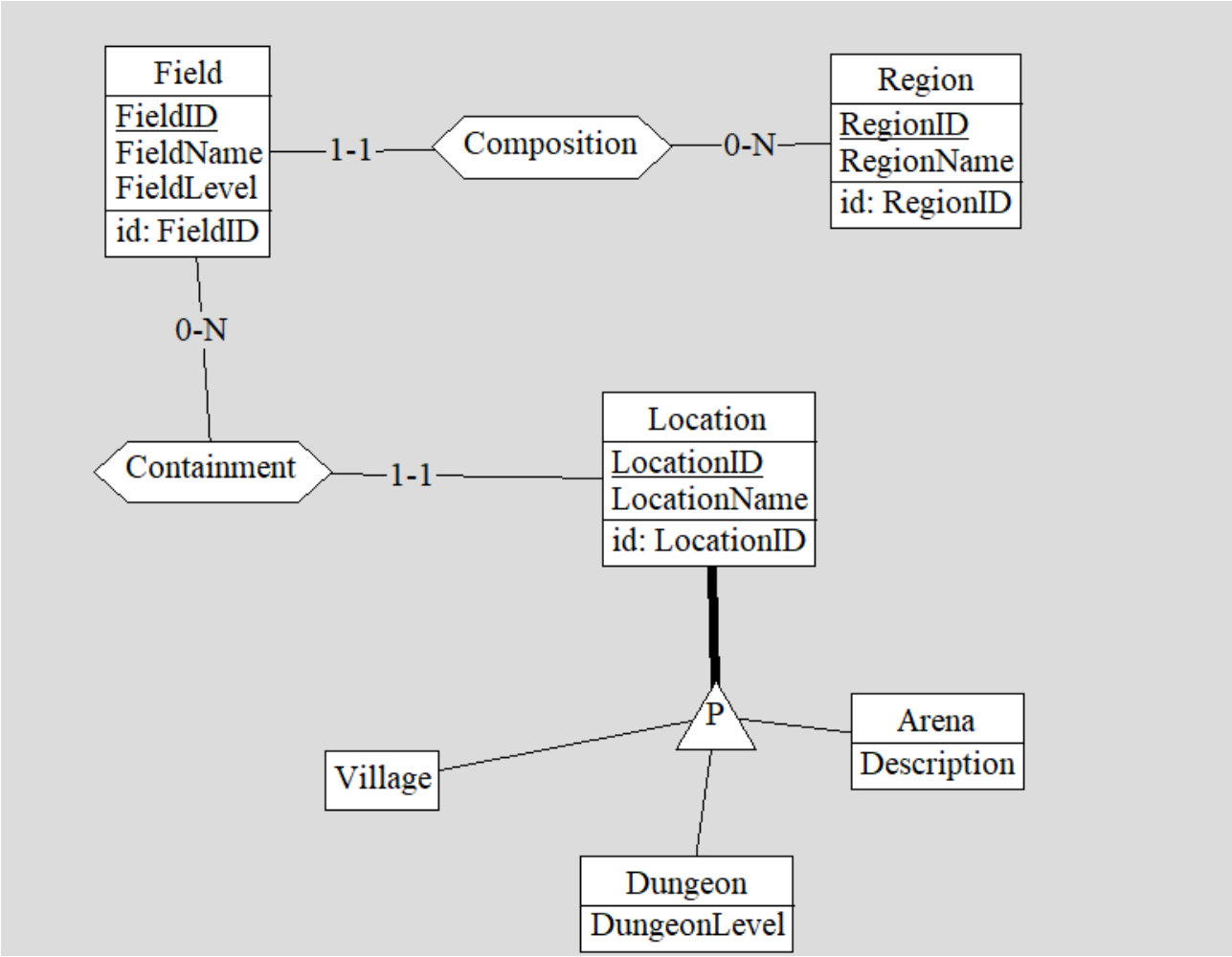
- Equipment (suddiviso in Weapon e Armor)
- QuestItem
- Material
- Currency
- Consumable

Tutti essi sono modellati da entità, e avranno come genitore l'entità Item, che conterrà campi comuni come nome e prezzo. Item è una entità centrale sia nel database, sia nel gioco. La raccolta degli item è infatti anche alla base del Gameplay, dunque sono molte le relazioni che collegano Item al resto delle parti del database. Innanzitutto, le già citate StackOfItem sono relative ad un certo tipo di Item e servono a modellare l'inventario dei giocatori. Le ricette di crafting richiedono degli Item come ingredienti e producono un Item come risultato (si veda l'NPC crafter e il sistema di crafting). Gli Item sono offerti come oggetti di vendita dai traders (si veda l'NPC trader e il sistema di trading). A volte è necessario raccogliere alcuni oggetti, di solito dei QuestItem, per completare delle quest. Gli item sono ottenuti come ricompensa dopo aver completato una Quest. Ogni nemico ha una percentuale particolare di droppare certi Item (per drop si intende rilasciare oggetti come "ricompensa" per aver sconfitto il suddetto nemico). Lo schema E-R risulterà come segue (le relazioni con gli NPC verranno approfondite in seguito)



3.4 - Sviluppo dell'ambito "World"

Si vuole anche modellare il mondo di gioco attraverso il sistema informativo, questo in previsione del fatto che in futuro potrebbe essere necessario raccogliere statistiche sulle zone più visitate, ma anche per aver un metodo flessibile per l'aggiunta o la modifica delle zone del mondo esistente. In generale, il mondo è composto da regioni. Le regioni contengono alcune zone (Field). Le zone contengono uno o più luoghi di interesse (Location), che possono essere villaggi, arene o dungeon. Nel database non vengono precisamente modellati i collegamenti tra una zona e l'altra (ad esempio, per sapere che strada occorre fare per arrivare da un luogo d'interesse A ad un luogo B). È più facile implementare questi algoritmi di pathfinding direttamente all'interno del gioco, anche perché il volume di dati delle tabelle relative a questo ambito è previsto essere sufficientemente basso.



3.5 - Sviluppo dell'ambito "NPC"

Gli NPC (Non-Playable Character) sono gli abitanti di un Village. Ogni Village ha i suoi abitanti. Ogni NPC ha un ruolo preciso, in generale aiutano il giocatore grazie al loro mestiere. Andiamo ora a elencare uno alla volta il ruolo e la scelta progettuale in termini di E-R per ogni tipo di NPC.

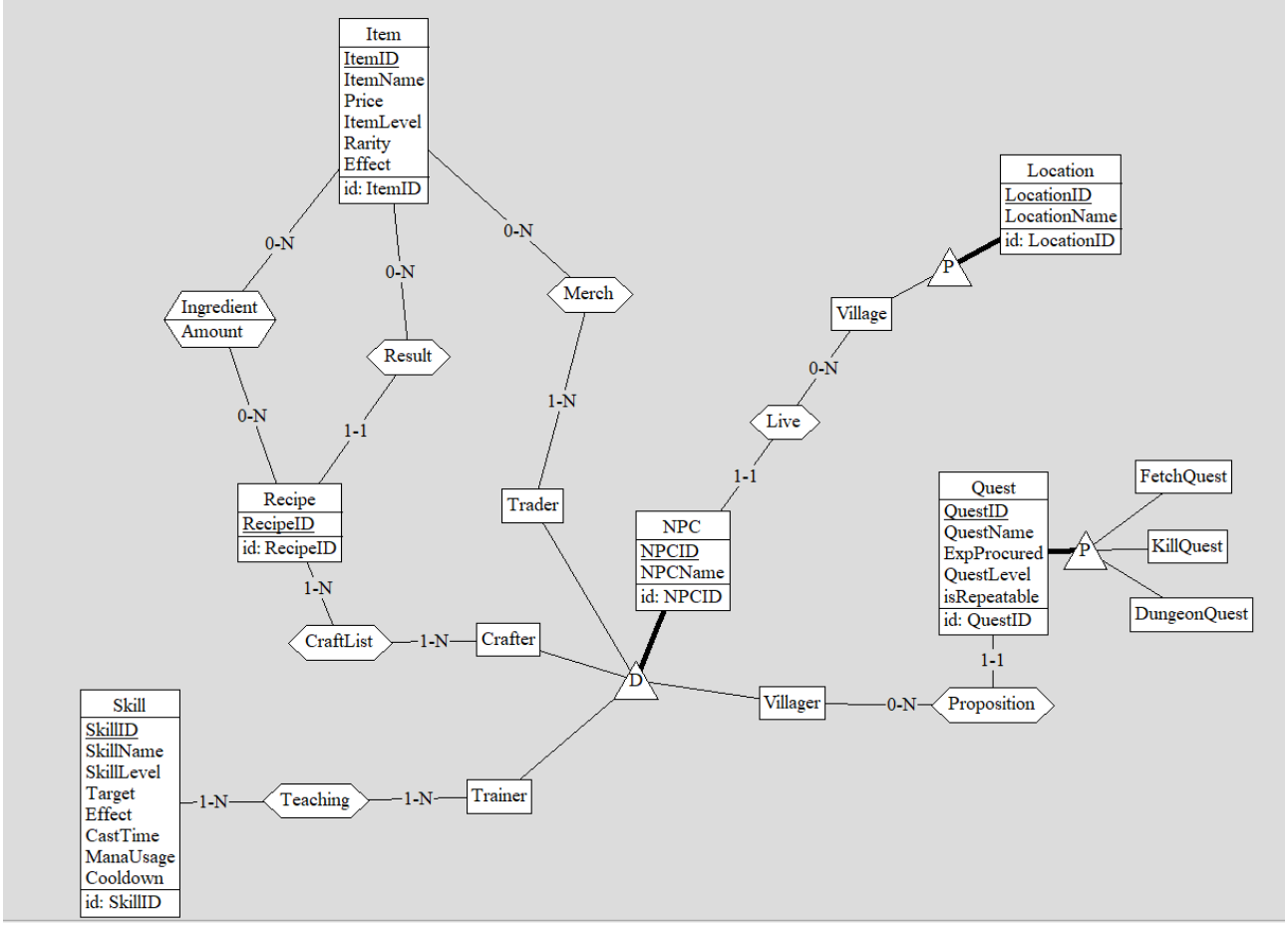
Trader: si tratta di commercianti. Offrono beni in cambio di denaro. Ogni trader ha un suo set di oggetti venduti (come se avesse una bancarella). Qui la scelta di modellazione è molto semplice, ovvero una relazione Merch che collega il Trader agli Item che vuole vendere. Il prezzo è una proprietà dell'Item, ciò significa che Trader diversi vendono lo stesso Item allo stesso prezzo.

Crafter: si tratta di alchimisti, fabbri, capomastri o altre figure di rilievo. In generale, producono oggetti complessi a partire dalle materie prime necessarie. Ad esempio un alchimista crea pozioni a partire da erbe e fiori, mentre un fabbro costruisce spade e armature a partire dai lingotti. In altre parole, ogni crafter conosce le ricette per produrre solo alcuni oggetti. Per modellare ciò, si avrà bisogno della relazione Recipe, che associa ad una lista di ingredienti/materiali il prodotto completo. Ogni crafter conoscerà solo alcune di queste Recipe (in maniera analoga a come un trader vende solo alcuni oggetti). Va detto che lo stesso Item può essere prodotto da più Recipe diverse, questo perché alcuni Crafters sono in realtà una sorta di barattieri, ovvero scambiano Currency per il prodotto finito (ad esempio, scambiare 100 gemme magiche per una spada d'oro anziché chiedere a un crafter diverso di costruire la spada a partire dai lingotti).

Trainer: permettono al Character di apprendere nuove Skill, se il livello del giocatore supera una certa soglia. Chiaramente non è detto che ogni trainer sappia insegnare tutte le skill esistenti.

Villager: semplici abitanti del villaggio, alcuni non fanno niente di particolare se non intrattenere una conversazione predefinita col giocatore. Altri invece possono proporre Quest, ovvero missioni. Come già accennato, esse si dividono in tre categorie (Fetch, Kill e Dungeon) e una volta completate, si verrà ricompensati dal Villager che ha richiesto aiuto. (si veda lo sviluppo dell'ambito "Game" per maggiori informazioni riguardo il sistema di quest)

Lo schema E-R di seguito illustrerà come si collegano gli NPC al resto dello schema.

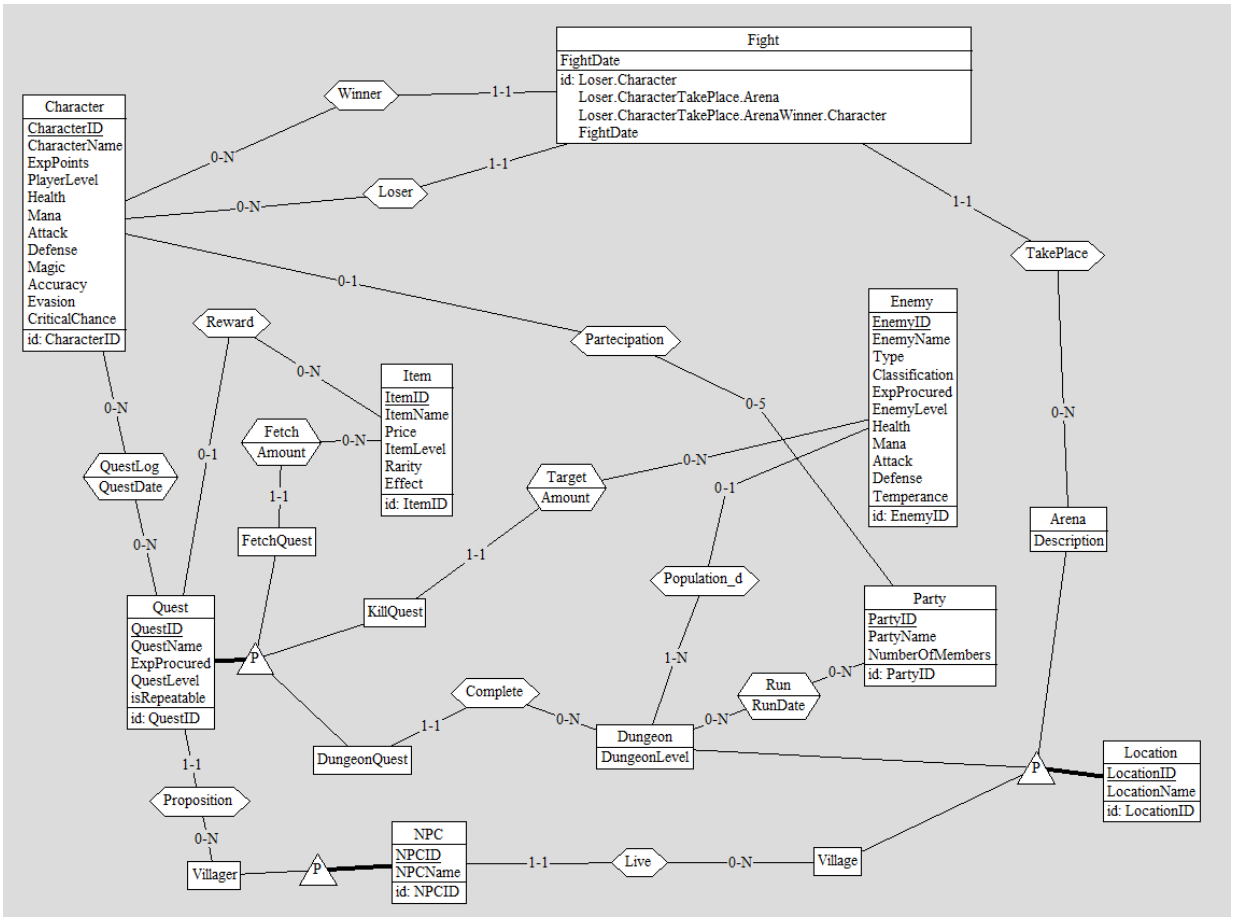


3.6 - Sviluppo dell'ambito "Game"

In quest'ambito si modellano tutte le entità e relazioni strettamente legate alle partite, ovvero le Quest (missioni giocatore singolo), le Run (avventure nei dungeon) e le Fight (duelli tra due giocatori). Soffermiamoci prima sulle quest. Esse sono offerte dagli NPC generici del villaggio, possono essere di tre tipi (FetchQuest, KillQuest e DungeonQuest), e offrono un oggetto e del denaro come ricompensa. Le quest meno importanti possono essere rigiocate più volte, mentre altre solo una volta. Si avrà bisogno dell'entità Quest (eventualmente genitore dei vari sottotipi di quest), legata al Villager con la relazione Offer, e una relazione QuestLog che collega Quest a Character, così che si possa tener traccia delle quest completate da ciascun character. Il fatto che le quest importanti siano giocabili una volta sola è un vincolo che verrà fatto rispettare eseguendo opportuni controlli a livello di gioco.

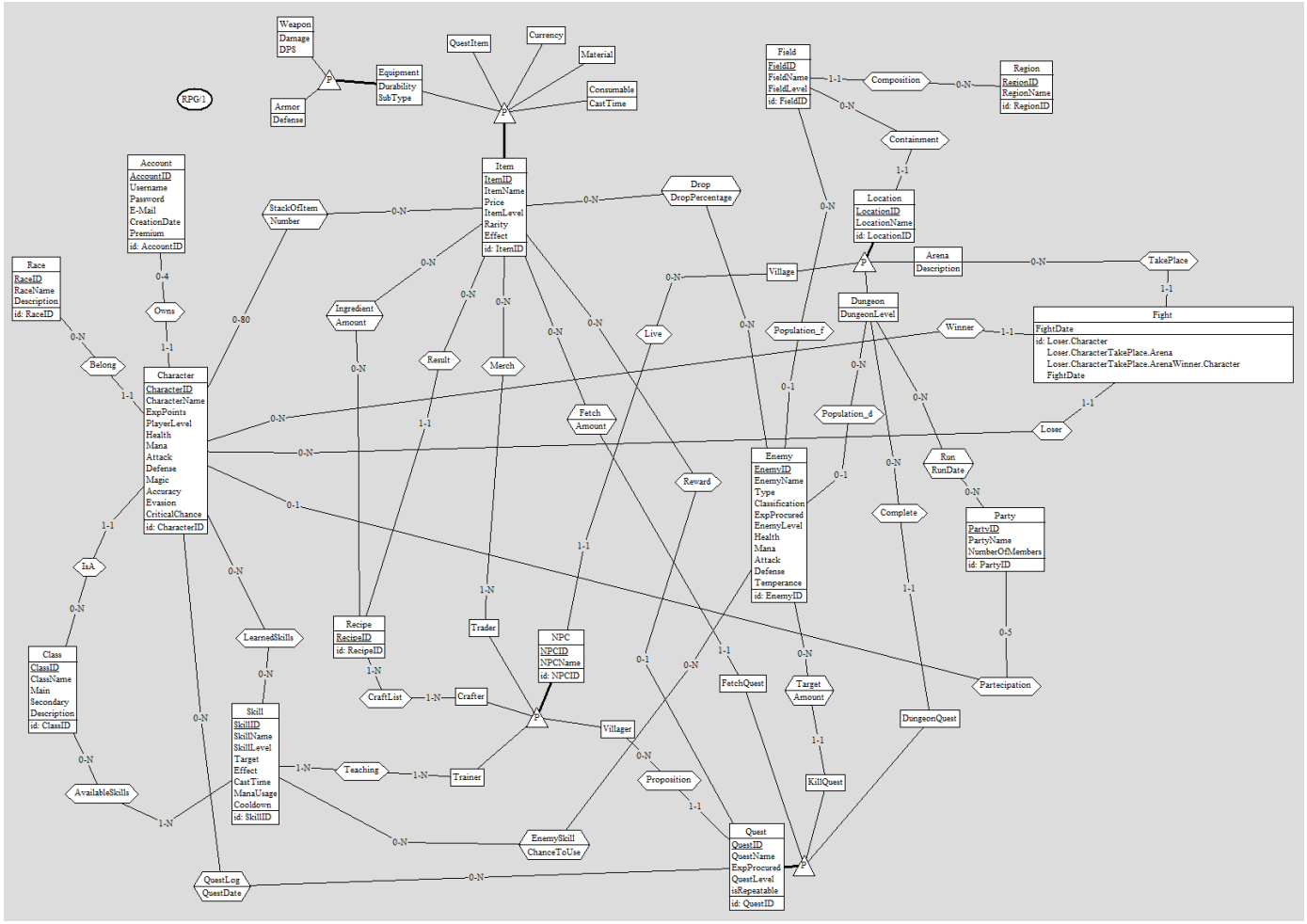
Altro tipo di partita sono le Run. Semplicemente una Run è una relazione come un QuestLog che collega un Party al dungeon che ha affrontato, per tener traccia delle partite effettuate. Ciò implica che i dungeon possono essere affrontati solo dai party e non dai giocatori singoli, tuttavia nulla vieta ad un giocatore di fondare un party di cui egli stesso è l'unico membro, se vuole affrontare il dungeon da solo.

Per le Fight la questione è leggermente più complessa. Le fight sono duelli disputati in un'arena, perciò per ogni partita si vuole risalire a chi ha combattuto e dove. Fight sarà perciò una entità con tre relazioni. Una relazione che permette di risalire al luogo dell'incontro, altre due che si collegano a Character e determinano il personaggio vincitore e il personaggio sconfitto. Per com'è modellato nello schema E-R, si potrebbe pensare di inserire dati inconsistenti nelle Fight, ad esempio il vincitore e lo sconfitto sono lo stesso giocatore. Questo però è impedito dal sistema di matchmaking del gioco stesso, che abbina sempre tra loro due giocatori diversi.



3.7 Anteprima dello schema generale

Ecco come appare lo schema generale del database, collegando tra loro i vari ambiti. Sono presenti anche alcune relazioni che fungono da “ponte” tra i vari ambiti che non erano state menzionate prima, nei singoli ambiti.



4 - SPECIFICHE FUNZIONALI

4.1 – Analisi delle funzionalità richieste

Durante la prima fase di progettazione del progetto logico, in cui lo schema concettuale verrà modificato in base a considerazioni statistiche sul carico di lavoro, sarà di fondamentale importanza rivolgere l'attenzione a un certo numero di operazioni sui dati necessarie a svolgere le funzionalità principali del sistema. Prendiamo ora in esame le operazioni elementari previste per il funzionamento del sistema e per la manipolazione dei dati, individuando per ognuna di esse i campi coinvolti e le informazioni da estrarre. Le principali funzionalità previste sono le seguenti:

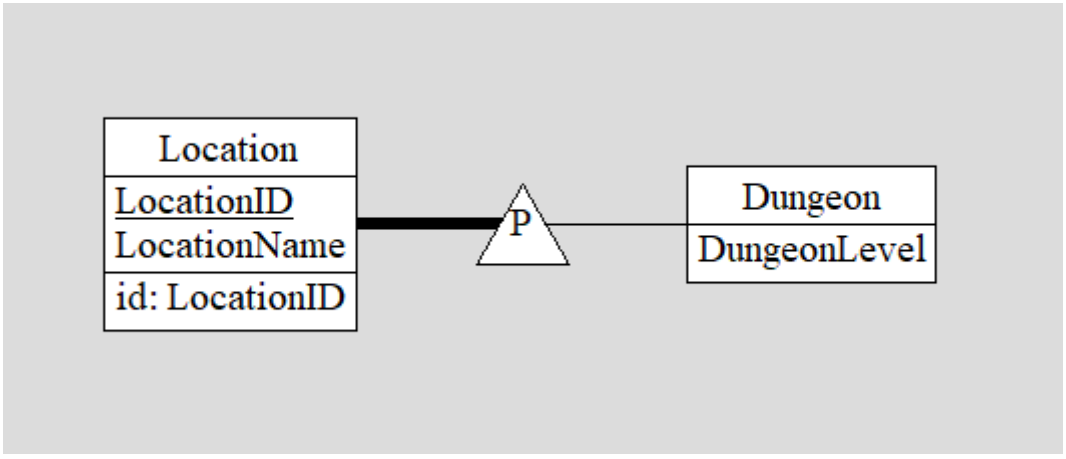
- Aggiunta e modifica di elementi relativi al mondo di gioco (amministratori)
- Visualizzare dati per futuri bilanciamenti delle classi (amministratori)
- Visualizzare i migliori giocatori per ciascun'arena (amministratori e giocatori)
- Visualizzare gli ultimi giocatori con i quali si è combattuto (giocatori)
- Visualizzare i luoghi in cui si può apprendere una determinata abilità (giocatori)
- Modi di ottenimento di un oggetto (giocatori)
- NPC presenti in un determinato villaggio (giocatori)

4.2 – Aggiunta e modifica di elementi relativi al mondo di gioco

Le operazioni relative all'ambito di aggiunta/modifica sono tante, ma anche molto simili tra loro; per questo si è deciso di portare una sola di aggiunta che valga da esempio per tutte le altre.
L'operazione scelta è l'aggiunta di un nuovo Dungeon.

Aggiunta di un nuovo Dungeon

ENTITA' COINVOLTE: Dungeon



4.3 – Visualizzare dati per futuri bilanciamenti delle classi

Questa operazione è di competenza esclusiva dell'amministratore, che deve poter visualizzare diverse statistiche calcolate sulla situazione attuale del gioco utili per futuri bilanciamenti. Le operazioni che compongono questa procedura sono 2:

- Visualizzazione del livello medio per ciascuna classe
- Visualizzazione delle classi più vincenti nei duelli

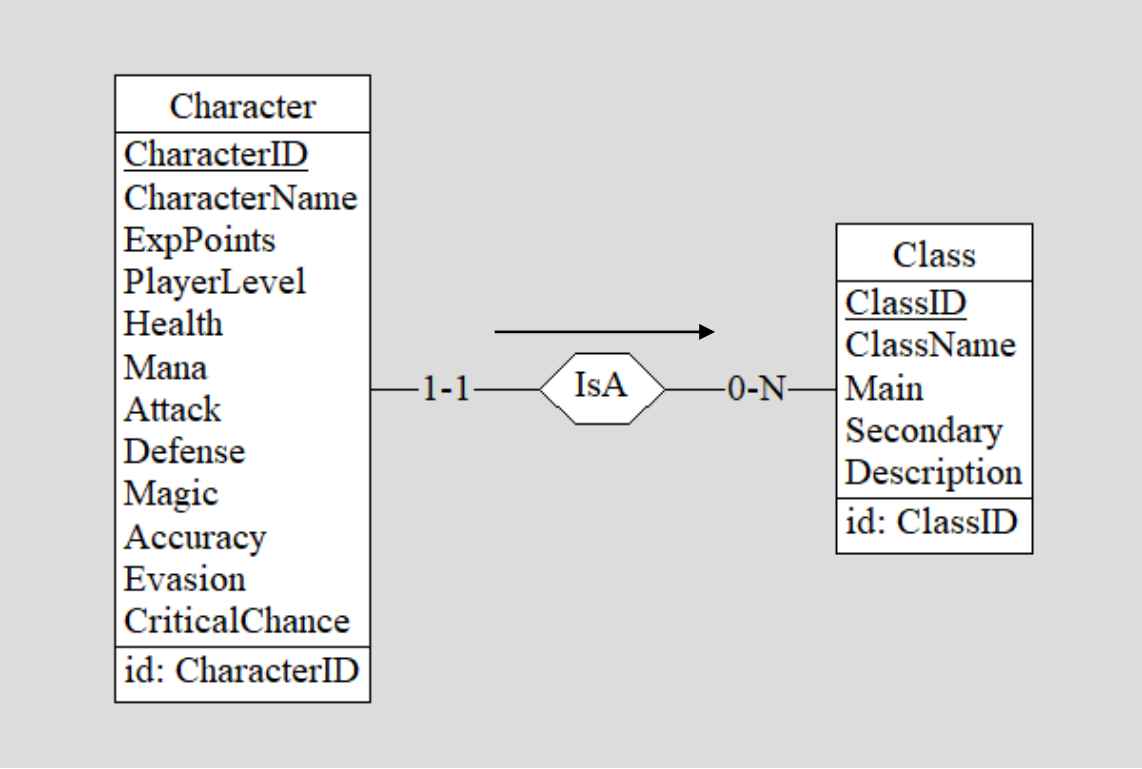
Nella visualizzazione del livello medio per classe vengono esclusi i personaggi di livello 0, ovvero quei personaggi che sono stati creati ma non effettivamente utilizzati nel gioco. Quest'operazione serve agli amministratori per controllare se ci sono classi con un livello medio molto più basso rispetto alle altre (magari perché sono più deboli o meno divertenti da giocare) e allo stesso tempo per valutare se il livello massimo raggiungibile è adeguato o è troppo facile da raggiungere. Questa procedura comporta la scansione di tutti i personaggi, i quali verranno raggruppati per classe; successivamente verrà calcolata la media su ciascun gruppo.

Nella visualizzazione delle classi più vincenti si contano le vittorie in duello per ogni classe; questo fornisce agli sviluppatori informazioni sulle classi preferite per i duelli. Questa procedura comporta la scansione di tutti i duelli con il conteggio delle vittorie ottenute dai vari personaggi, poi raggruppate per classe.

Visualizzazione del livello medio per ciascuna classe:

ENTITA' COINVOLTE: Character, Class

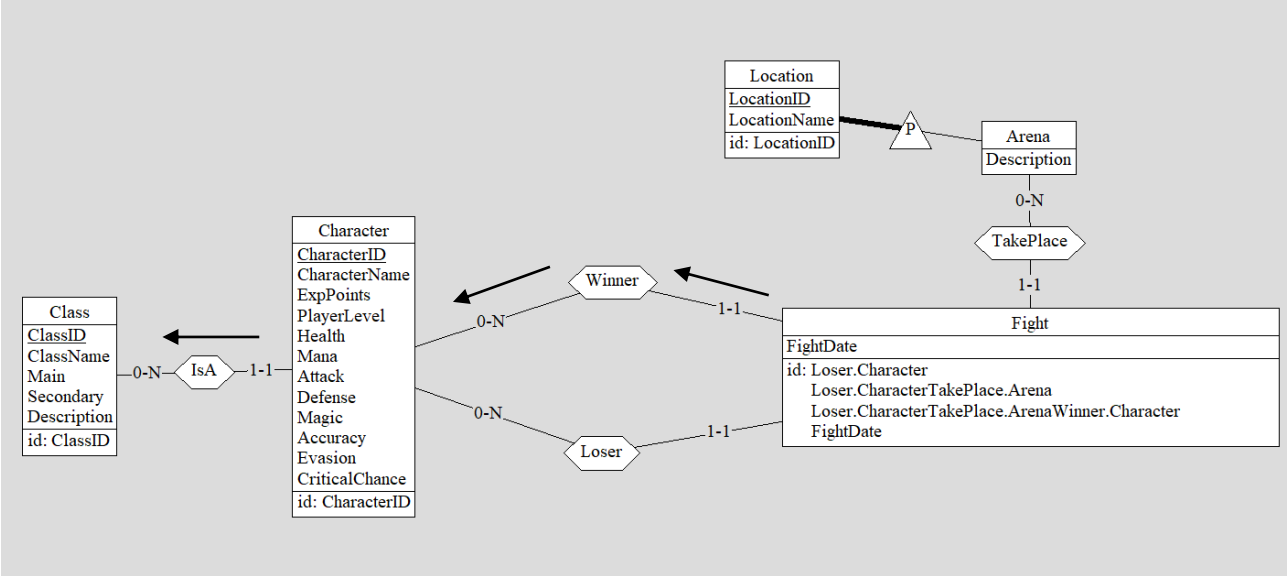
ASSOCIAZIONI COINVOLTE: IsA



Visualizzazione delle classi più vincenti nei duelli

ENTITA' COINVOLTE: Character, Class, Fight

ASSOCIAZIONI COINVOLTE: IsA, Winner



4.4 – Visualizzare i migliori giocatori per ogni Arena

Questa operazione è utilizzabile allo stesso modo da amministratori e giocatori; mentre ai primi è necessaria per conferire premi ai personaggi più meritevoli, agli utenti è utile per vedere se compaiono nella leaderboard (il che non ha alcuna valenza pratica al di fuori dell'autogratificazione). Le operazioni che compongono questa procedura sono 2:

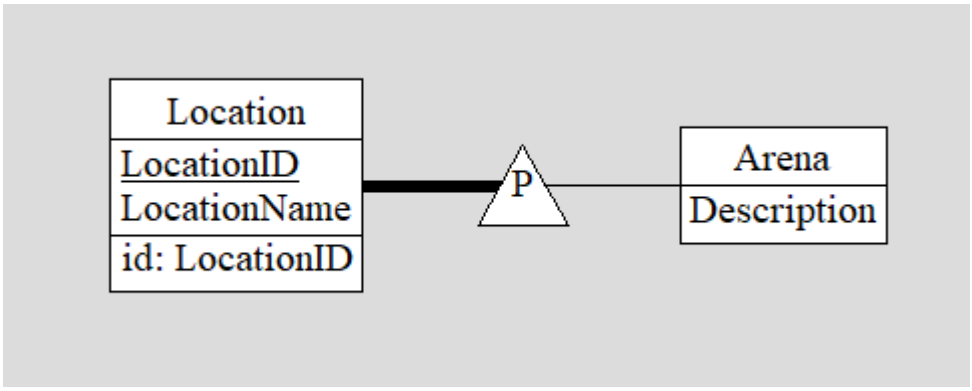
- Selezione dell'arena specificata
- Visualizzazione dei giocatori con più vittorie in tale arena

L'azione di selezione dell'arena specificata consiste nel trovare l'arena che corrisponde a quella cercata dall'utente (sia esso amministratore o giocatore).

Nella visualizzazione dei giocatori con più vittorie in tale arena si contano le vittorie che ciascun personaggio ha conseguito nell'arena scelta. Quest'operazione comporta la scansione delle battaglie svolte nell'arena scelta con il conteggio delle vittorie di ogni personaggio, poi visualizzate in ordine decrescente.

Selezione dell'arena specificata:

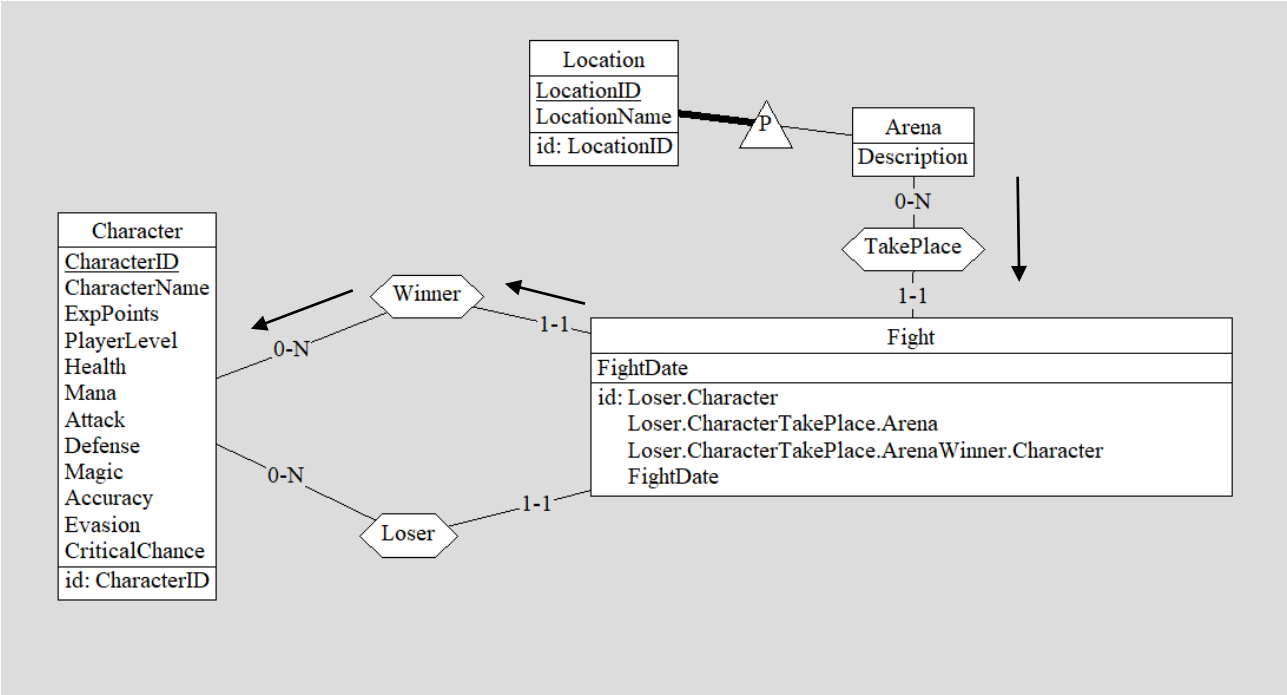
ENTITA' COINVOLTE: Arena



Visualizzazione dei giocatori con più vittorie in tale arena:

ENTITA' COINVOLTE: Arena, Character, Fight

ASSOCIAZIONI COINVOLTE: TakePlace, Winner



4.5 – Visualizzare gli ultimi giocatori con cui si è combattuto

Questa operazione è utilizzabile esclusivamente dai giocatori, che in questo modo potranno aggiungere gli ultimi avversari incontrati in combattimento alla "lista amici" o alla "lista bloccati", a seconda dell'interazione avuta. Quest'operazione si compone di una sola operazione:

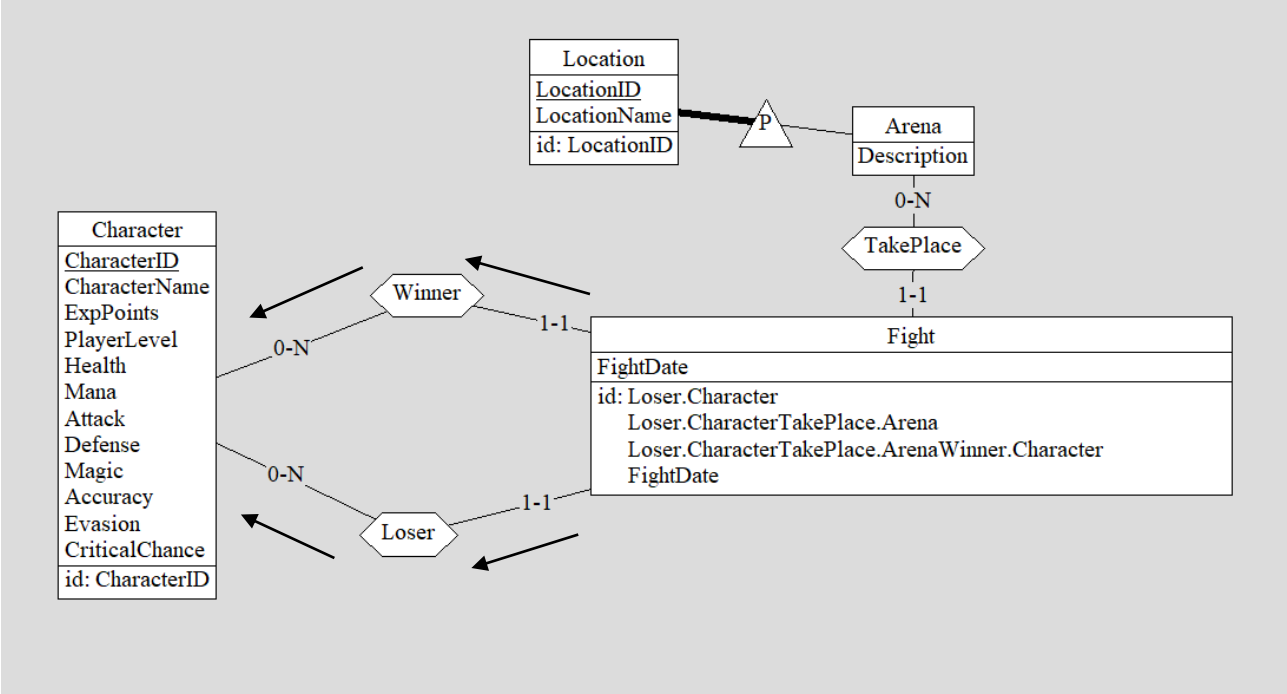
- Visualizzazione degli ultimi personaggi con cui si è combattuto

Per svolgere l'operazione è innanzitutto necessario selezionare le ultime battaglie a cui il personaggio scelto ha partecipato, basandosi sulla data in cui questo è avvenuto; fatto ciò si è deciso di raggruppare le eventuali battaglie contro lo stesso personaggio, così da evitare situazioni nelle quali la lista risulta composta per la maggior parte da pochi avversari (magari perché già nella lista amici o per qualche meccanica di "rematch")

Visualizzazione degli ultimi personaggi con cui si è combattuto:

ENTITA' COINVOLTE: Character, Fight

ASSOCIAZIONI COINVOLTE: Winner, Loser



4.6 – Visualizzare i luoghi in cui si può apprendere una determinata abilità

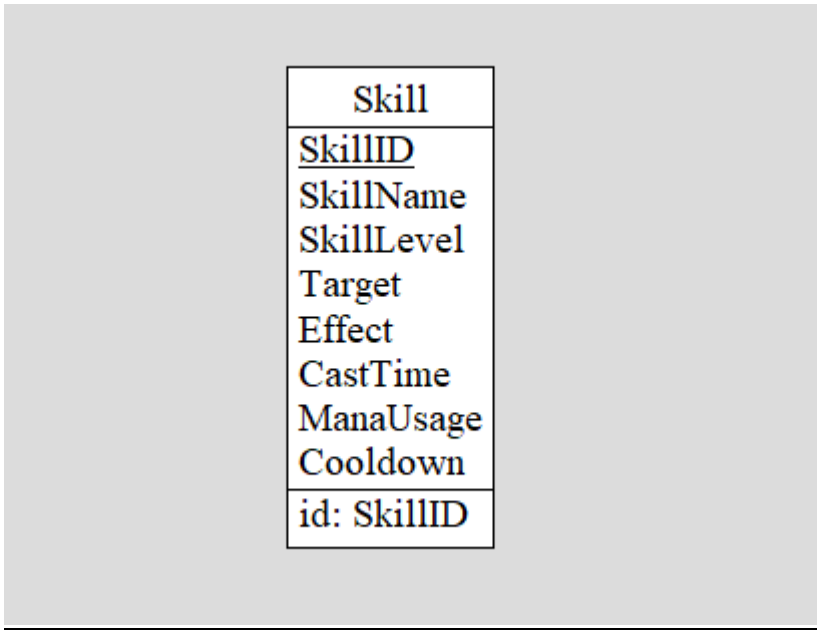
Questa operazione è utilizzabile dai giocatori, cosicché nel momento in cui si voglia apprendere una specifica abilità, si riesca a farlo senza la necessità di procedere per tentativi nei villaggi. Quest'operazione si compone di 2 fasi:

- Selezione dell'abilità che si vuole imparare
- Visualizzazione dei luoghi in cui è possibile impararla

Dopo la selezione dell'abilità che si vuole apprendere verranno mostrati gli allenatori in grado di insegnarla, con relativi nomi del villaggio in cui abita, della zona e della regione.

Selezione dell'abilità che si vuole imparare:

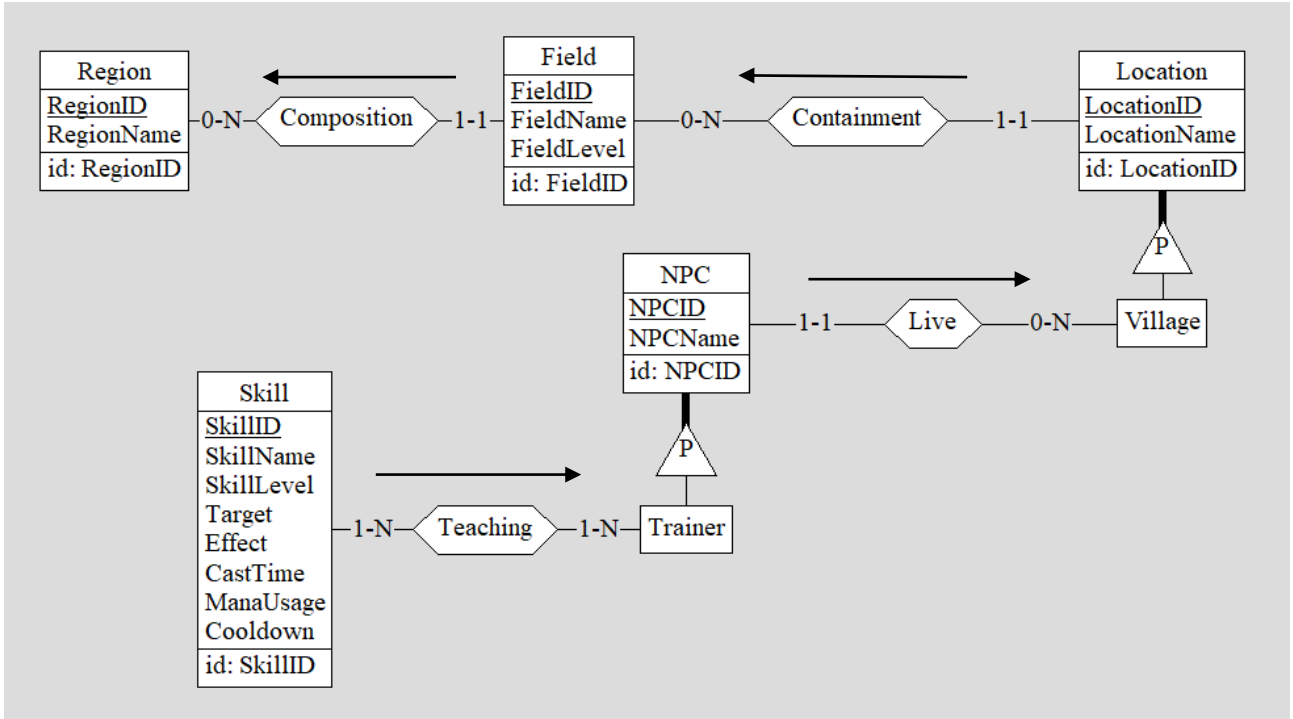
ENTITA' COINVOLTE: Skill



Visualizzazione dei luoghi in cui è possibile impararla:

ENTITA' COINVOLTE: Field, Region, Skill, Trainer, Village, Location, NPC

ASSOCIAZIONI COINVOLTE: Composition, Containment, Live, Teaching



4.7 – Modi di ottenimento di un oggetto

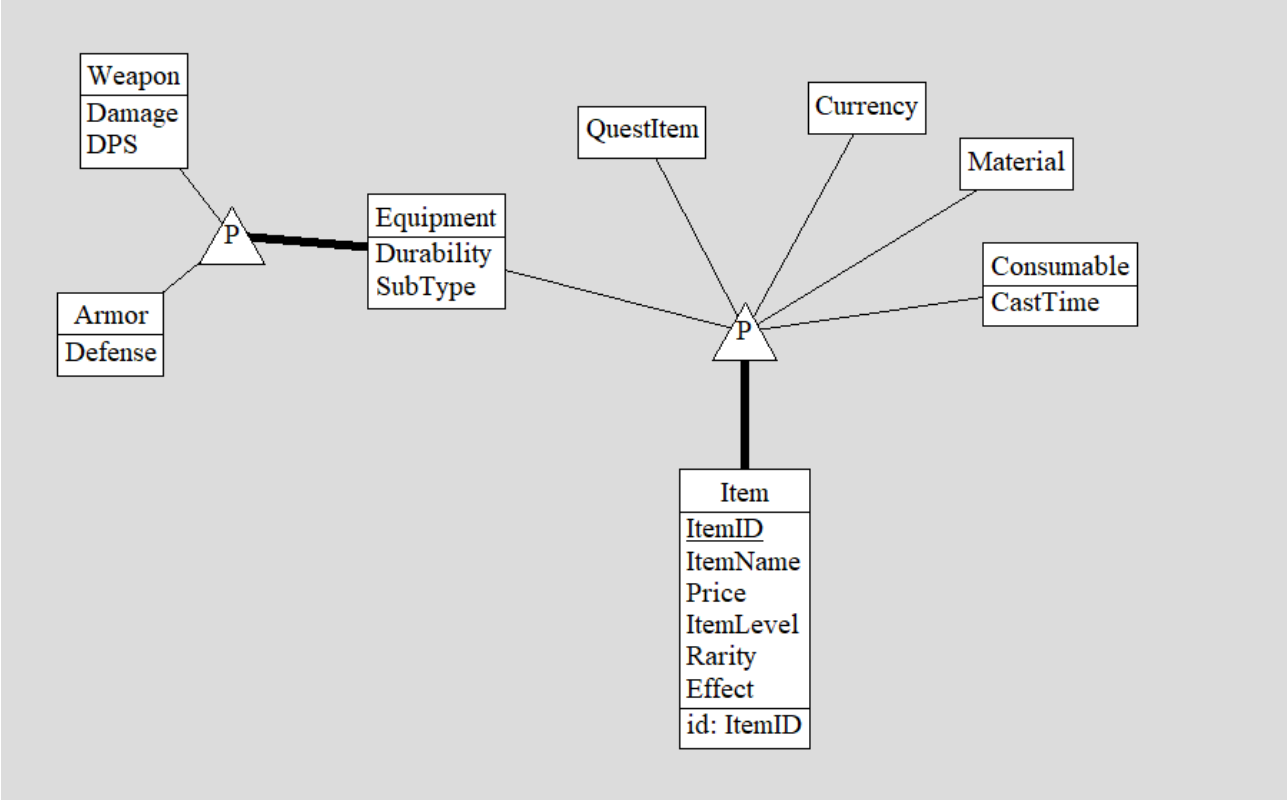
Questa operazione ha una valenza simile a quella precedente: può essere utile ad un giocatore in cerca di uno specifico oggetto, così da non dover procedere per tentativi. Dato che ci sono diversi modi di ottenere un oggetto (ricompensa di una missione, drop di un nemico, vendita di un commerciante…) la tabella risultante consiste nell'unione dei singoli modi, specificando tramite un apposito campo di quale modo si tratta (nemico, missione, commerciante…). Quest'operazione si compone di 2 fasi:

- Selezione dell'oggetto che si vuole ottenere
- Visualizzazione dei modi in cui si può ottenere

Per selezionare l'oggetto basta risalire all'ID dell'oggetto di cui si conosce il nome. Fatto ciò si dovranno visitare le varie entità che possono fornire un oggetto, selezionando quelle che forniscono ciò che si desidera, unendo infine i risultati.

Selezione dell'oggetto da ispezionare:

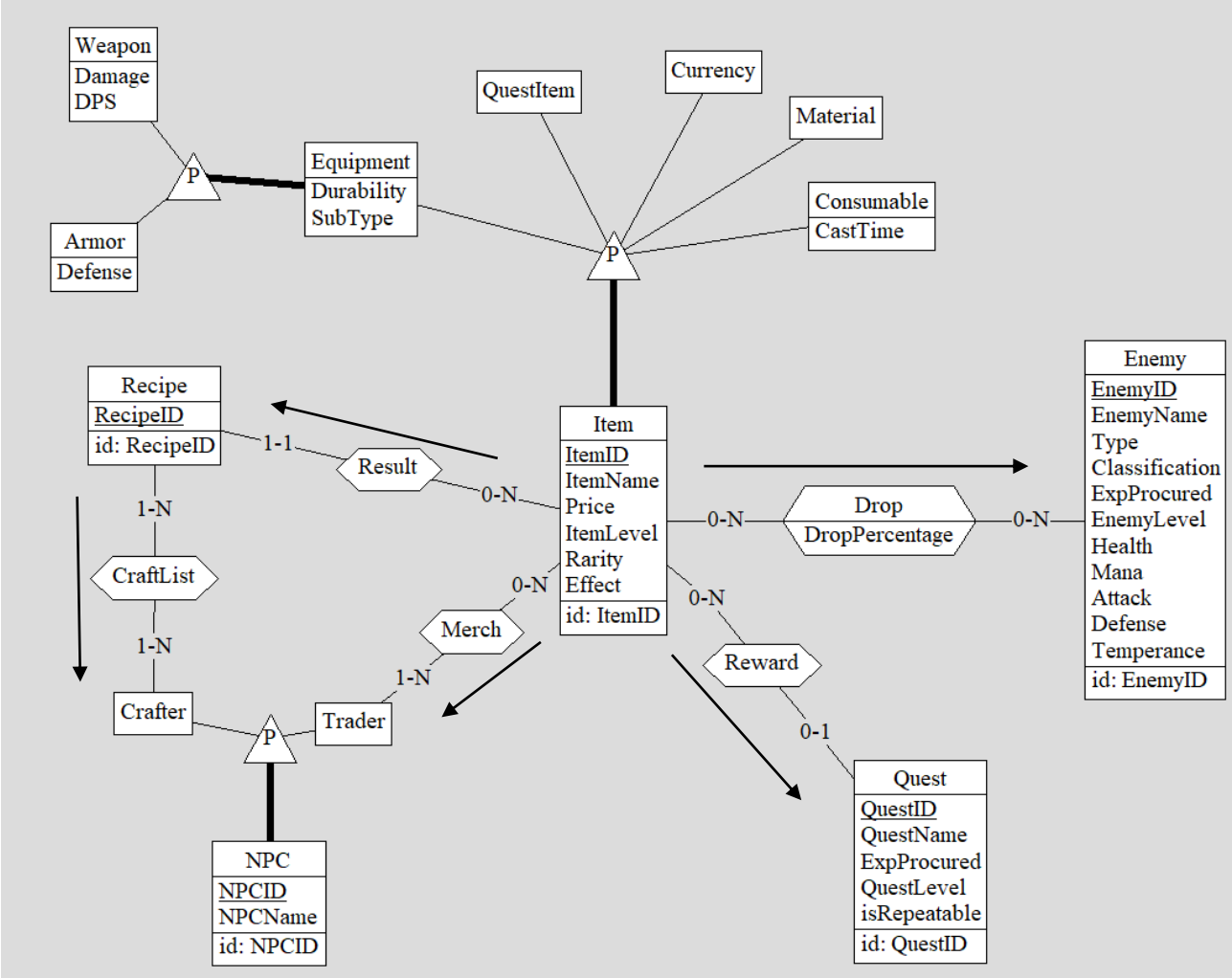
ENTITA' COINVOLTE: Item



Visualizzazione dei modi in cui si può ottenere:

ENTITA' COINVOLTE: Crafter, Enemy, Quest, Recipe, Trader, Item

ASSOCIAZIONI COINVOLTE: CraftList, Drop, Merch, Result, Reward



4.8 – NPC presenti in un determinato villaggio

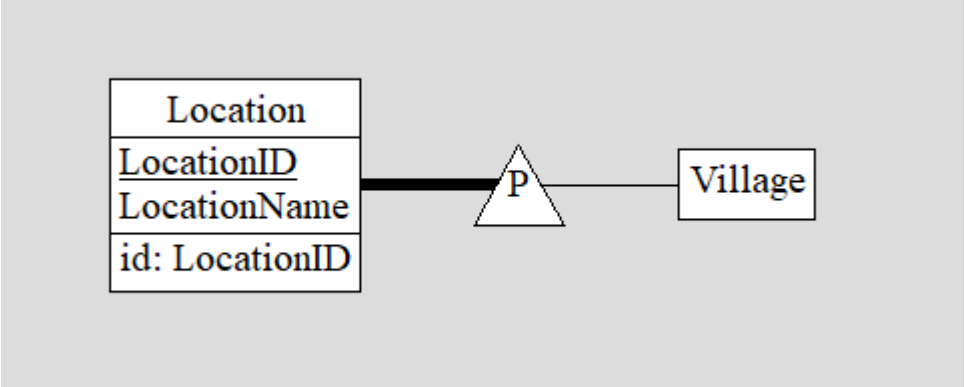
Quest'operazione serve ai giocatori per vedere la lista di tutti gli npc presenti in un villaggio; dato che in un villaggio coesistono vari tipi di npc, relativi a entità diverse, si fa uso del comando union per racchiudere in un'unica tabella il risultato finale. L'operazione si compone quindi di due fasi:

- Selezione del villaggio
- Visualizzazione degli npc del villaggio scelto

Dopo la selezione del villaggio si dovranno unire i vari tipi di npc che "abitano" in quel villaggio in un'unica tabella.

Selezione del villaggio:

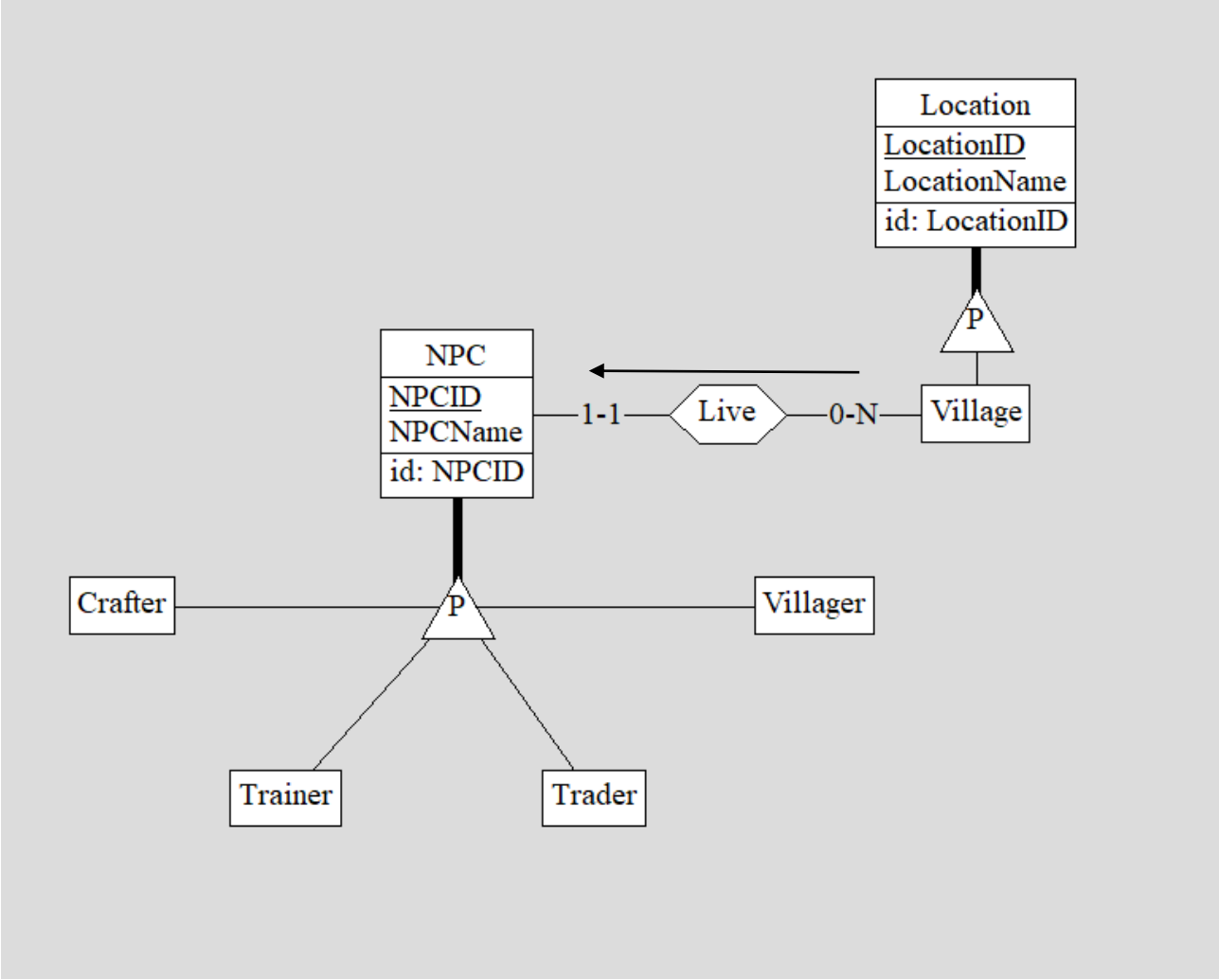
ENTITA' COINVOLTE: Village



Visualizzazione degli npc del villaggio scelto:

ENTITA' COINVOLTE: Crafter, Trader, Trainer, Village, Villager, NPC

ASSOCIAZIONI COINVOLTE: Live



CARICHI DI LAVORO

La tabella riportata di seguito contiene il volume di dati che occupa (mediamente) le varie istanze delle entità e associazioni individuate dal nostro schema concettuale. In base a tali valori, come già accennato, lo schema verrà in un primo momento rimodellato per garantire una migliore efficienza in rapporto alle principali funzionalità che abbiamo descritto, e successivamente sarà ritoccato per aderire meglio alle specifiche imposte dal particolare modello logico con il quale rappresenteremo i dati.

NOME	TIPO	VOLUME DI DATI
ACCOUNT	E	1000
ARENA	E	10
ARMOR	E	300
AVAILABLE SKILL	R	300
BELONG	R	3000
CHARACTER	E	3000
CLASS	E	10
COMPLETE	R	40
COMPOSITION	R	40
CONTAINMENT	R	50
CONSUMABLE	E	30
CRAFTER	E	20
CRAFT LIST	R	100
CURRENCY	E	10
DROP	R	500
DUNGEON	E	20
DUNGEON QUEST	E	40
ENEMY	E	150
ENEMY SKILL	R	400
EQUIPMENT	E	1000
FETCH	R	40
FETCH QUEST	E	40
FIELD	E	40
FIGHT	E	3000000
INGREDIENT	R	200
IS A	R	3000
ITEM	E	3000
KILL QUEST	E	40
LEARNED SKILL	R	28000
LOCATION	E	50
LOSER	R	3000000
LIVE	R	150
MATERIAL	E	2000
MERCH	R	300
NPC	E	150
OWNS	R	3000
PARTECIPATION	R	2500
PARTY	E	250
POPULATION_D	R	120
POPULATION_F	R	50

PROPOSITION	R	100
QUEST	E	100
QUEST ITEM	E	20
QUEST LOG	R	150000
RACE	E	5
RECIPE	E	50
REGION	E	10
RESULT	R	50
RUN	R	1200
SKILL	E	150
STACK OF ITEM	R	150000
TAKE PLACE	R	3000000
TEACHING	R	100
TRADER	E	20
TRAINER	E	20
VILLAGE	E	20
VILLAGER	E	70
WEAPON	E	700
WINNER	R	3000000

5 – IL PROGETTO LOGICO

5.1 – Schemi di navigazione

Riassumiamo in breve le funzionalità principali richieste al sistema:

- Aggiunta e modifica di elementi relativi al mondo di gioco (amministratori)
- Visualizzare dati per futuri bilanciamenti delle classi (amministratori)
- Visualizzare i migliori giocatori per ciascun'arena (amministratori e giocatori)
- Visualizzare gli ultimi giocatori con i quali si è combattuto (giocatori)
- Visualizzare i luoghi in cui si può apprendere una determinata abilità (giocatori)
- Modi di ottenimento di un oggetto (giocatori)
- NPC presenti in un determinato villaggio (giocatori)

Per ciascuna di queste operazioni e per le operazioni elementari che le compongono individuiamo ora lo schema di navigazione E/R.

5.1.1 Aggiunta di un nuovo Dungeon

L'aggiunta di un nuovo Dungeon consiste nella semplice aggiunta di una nuova tupla all'entità Dungeon.

5.1.2 Visualizzare dati per futuri bilanciamenti

5.1.2.1 Visualizzazione del livello medio per ogni classe

Innanzitutto bisogna scandire tutte le istanze della relazione IsA così da raggruppare i character in base alla base alla classe, dopodiché si calcolerà il livello medio per ogni classe.

5.1.2.2 Visualizzazione delle classi più vincenti nei duelli

La visualizzazione delle classi più vincenti nei duelli avviene osservando per ogni fight qual è il character vincente, da questo ricavarne la classe a cui appartiene e infine sommando le vittorie in base alla classe.

5.1.3 Visualizzare i migliori giocatori per ciascun'arena

5.1.3.1 Selezione dell'arena specificata

Quest'operazione consiste nel selezionare un'arena tra quelle presenti nel gioco. Ciò comporta visualizzare ogni singola arena presente nel gioco. L'utente selezionerà dalla lista l'arena desiderata, che verrà usata come input per l'operazione seguente

5.1.3.2 Visualizzazione dei giocatori con più vittorie in tale arena

Dopo aver scelto l'arena, quest'operazione mostra i migliori due giocatori per essa, osservando tutte le Fight avvenute nella suddetta arena e contando le vittorie ottenuta da ogni Character.

5.1.4 Visualizzare gli ultimi giocatori con i quali si è combattuto

Quest'operazione consiste nel controllare le ultime 5 Fight a cui il nostro Character ha partecipato e nel visualizzare gli avversari. Questa operazione dovrà scorrere lo storico delle partite, selezionando quelle in cui il nostro Character ha partecipato come vincitore o come perdente.

5.1.5 Visualizzare i luoghi in cui si può apprendere una determinata abilità

5.1.5.1 Selezione dell'abilità che si vuole imparare

Quest'operazione consiste nel selezionare una Skill tra quelle presenti nel gioco. Ciò significa visualizzare ogni singola abilità presente per poterne selezionare una specifica da passare in input all'operazione seguente.

5.1.5.2 Visualizzazione dei luoghi in cui è possibile impararla

Quest'operazione consiste nel controllare quali Trainer insegnano la Skill precedentemente selezionata. Per fare ciò si dovrà non solo risalire a quali Trainer sono legati alla Skill selezionata, ma si dovrà risalire anche al luogo in cui vivono (Village, Field e Region).

5.1.6 Modi di ottenimento di un oggetto

5.1.6.1 Selezione dell'oggetto che si vuole ottenere

Questa operazione consiste nello scegliere un oggetto che verrà passato in input all'operazione successiva. Per fare ciò si dovranno scorrere uno per uno tutti gli Item esistenti.

5.1.6.2 Visualizzazione dei modi in cui si può ottenere

Per visualizzare l'elenco completo dei modi in cui è possibile ottenere un oggetto è necessario mettere insieme dati eterogenei. Ciò include:

- Cercare i Crafter che conoscono Recipe che producono l'oggetto desiderato
- Cercare gli Enemy che droppano l'oggetto richiesto
- Cercare i Trader che vendono l'oggetto voluto
- Cercare le Quest che offrono come premio l'oggetto scelto

5.1.7 NPC presenti in un determinato villaggio

5.1.7.1 Selezione del villaggio

Per selezionare il villaggio è necessario, come di consueto, scorrere tra tutti i Village esistenti tra i quali sarà scelto solo uno che verrà passato in input all'operazione successiva

5.1.7.2 Visualizzazione degli NPC del villaggio scelto

Per questa operazione è necessario unire tra loro dati provenienti dalle varie entità che modellano gli abitanti del villaggio. Essi sono i Trader, i Trainer, i Crafter e i Villager. Tra essi verranno selezionati solo quelli che vivono nel villaggio scelto.

5.2 – Frequenza e costo degli accessi

Funzionalità		Frequenza
Numero	Descrizione	
1	Aggiunta di un nuovo Dungeon	1/Mese
2	Visualizzare dati per futuri bilanciamenti delle classi	1/Mese
3	Visualizzare i migliori giocatori per ciascun'arena	1/Mese
4	Visualizzare gli ultimi giocatori con i quali si è combattuto	1000/Giorno
5	Visualizzare i luoghi in cui si può apprendere una determinata abilità	500/Giorno
6	Modi di ottenimento di un oggetto	500/Giorno
7	NPC presenti in un determinato villaggio	100/Giorno

La tabella qui sopra mostra la frequenza con la quale si farà uso delle operazioni presentate nella sezione precedente. Questi dati ci consentiranno di estrapolare, per ogni operazione elementare in cui abbiamo visto scomposte le suddette procedure, il volume degli accessi e il carico di lavoro. I calcoli appena accennati sono riportati nella tabella seguente.

TABELLA DEGLI ACCESSI

Operazione	Concetto	Entità/Assoc.	Lettura/Scrittura	Numero di accessi	Totale degli accessi (al mese)
1 – Aggiunta di un nuovo dungeon	Run	E	S	1	1
2.1 - Visualizzazione del livello medio per ogni classe	Character	E	L	3000	3000
	Class	E	L	10	10
	IsA	A	L	3000	3000
2.2 – Visualizzazione delle classi più vincenti nei duelli	Character	E	L	3000	3000
	Class	E	L	10	10
	Fight	E	L	3000000	3000000
	IsA	A	L	3000	3000
	Winner	A	L	3000000	3000000
3.1 – Selezione dell'arena specificata	Arena	E	L	10	10
3.2 – Visualizzazione dei giocatori con più vittorie in tale arena	Arena	E	L	1	1
	Character	E	L	2500	2500
	Fight	E	L	300000	300000
	TakePlace	A	L	300000	300000
	Winner	A	L	2500	2500
4 – Visualizzazione degli ultimi personaggi con cui si è combattuto	Character	E	L	1000	30000000
	Fight	E	L	5000	150000000
	Winner	A	L	5000	150000000
	Loser	A	L	5000	150000000
5.1 – Selezione dell'abilità che si vuole imparare	Skill	E	L	150	2250000
5.2 – Visualizzazione dei luoghi in cui è possibile impararla	Field	E	L	5	2500
	Region	E	L	5	2500
	Skill	E	L	1	500
	Trainer	E	L	5	2500
	Village	E	L	5	2500

	Location	E	L	5	2500
	NPC	E	L	5	2500
	Composition	A	L	5	2500
	Containment	A	L	5	2500
	Live	A	L	5	2500
	Teaching	A	L	100	50000
6.1 – Selezione dell'oggetto da ispezionare	Item	E	L	3000	1500000
6.2 – Visualizzazione dei modi in cui si può ottenere	Crafter	E	L	1	500
	Enemy	E	L	3	1500
	Quest	E	L	2	1000
	Recipe	E	L	1	500
	Trader	E	L	3	1500
	Item	E	L	1	500
	CraftList	A	L	100	50000
	Drop	A	L	500	250000
	Merch	A	L	300	150000
	Result	A	L	50	25000
	Reward	A	L	100	50000
7.1 – Selezione del villaggio	Village	E	L	20	2000
7.2 – Visualizzazione degli NPC del villaggio scelto	Crafter	E	L	2	200
	Trader	E	L	2	200
	Trainer	E	L	2	200
	Village	E	L	1	100
	Villager	E	L	3	300
	NPC	E	L	8	800
	Live	A	L	150	15000

5.3 – Trasformazioni dello schema concettuale

In questa sezione ci occupiamo delle ultime trasformazioni che applicheremo al nostro schema concettuale prima di dedicarci al suo ritocco in vista del particolare modello logico con cui sceglieremo di rappresentare la nostra realtà.

Si tratta per lo più di scegliere le chiavi primarie per ogni entità, di rimuovere quegli aspetti generalmente non supportati dai modelli logici, come le gerarchie di generalizzazione. C'è da precisare che viene tralasciata la parte sull'analisi delle ridondanze perché abbiamo ritenuto avessero poco senso all'interno delle operazioni che abbiamo preso in considerazione; detto questo è comunque presente un'unica ridondanza nell'entità party che mantiene in proprio una copia del numero di membri che lo compongono, ciò in previsione di operazioni di aggiunta di un character al party, in modo da facilitarne il controllo sul fatto che quest'ultimo sia già pieno. Procedendo step-by-step, quindi:

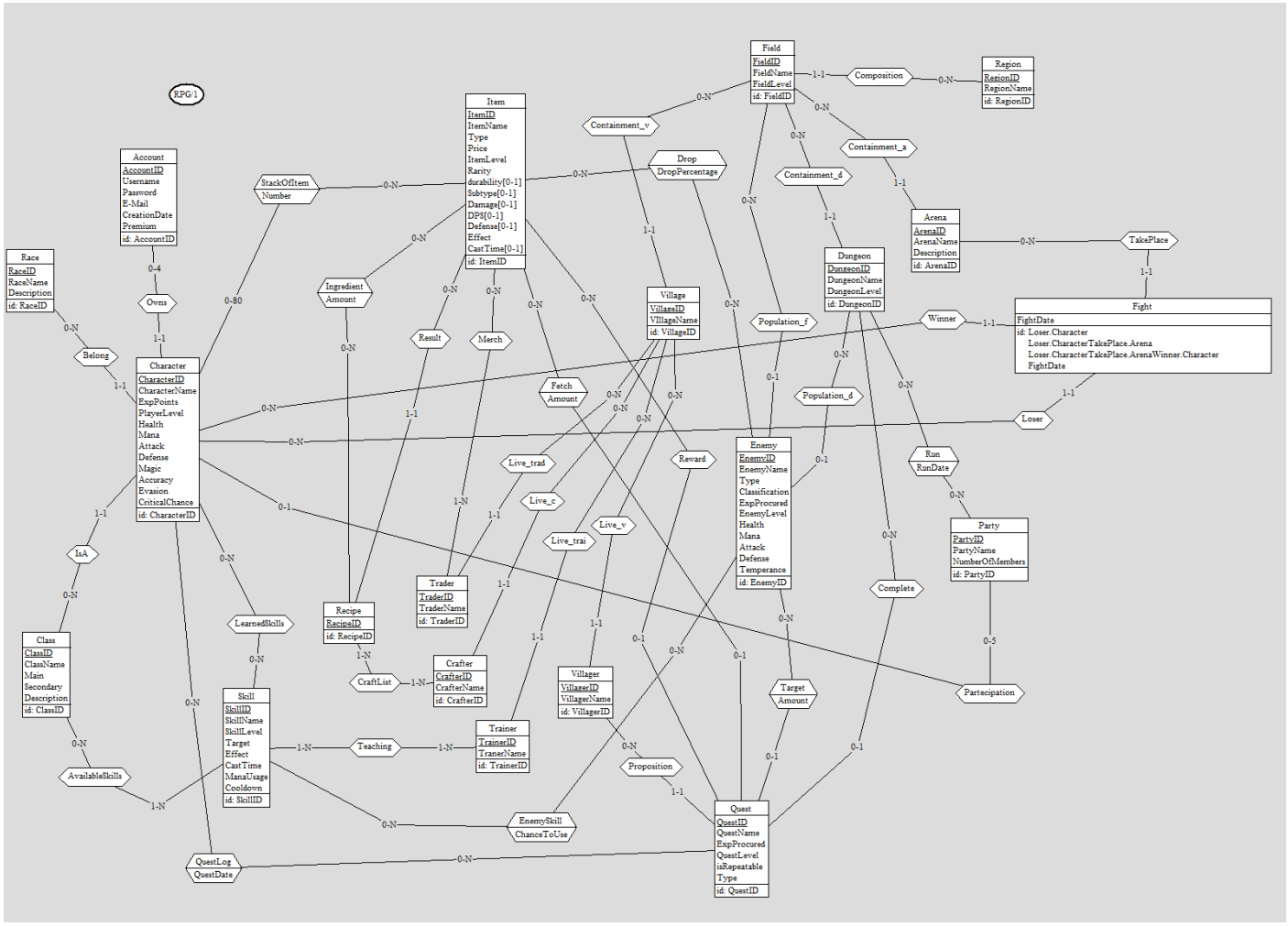
5.3.1 – Scelta delle chiavi primarie

La scelta delle chiavi primarie è un'operazione banale, dal momento che ogni entità (e non poteva essere altrimenti, vista la natura del nostro sistema informativo) è univocamente identificata da un campo numerico.

5.3.2 – Eliminazione delle gerarchie di generalizzazione

Nel nostro schema concettuale le generalizzazioni da fare sono molteplici. Procediamo con ordine:

- La generalizzazione dell'entità Quest, di tipo totale ed esclusiva. Abbiamo scelto di far collassare verso l'alto; questa scelta è stata dettata dalla presenza dell'associazione QuestLog, che se avessimo scelto il collasso verso il basso sarebbe dovuto essere diviso in tre (uno per ogni sottotipo di quest). Pur vero che con questa soluzione c'è il bisogno di introdurre campi che hanno tra loro una relazione di or (ossia se un campo ha un valore, sapremmo per certo che altri due valori siano NULL) abbiamo comunque ritenuto che fosse il modo più comodo di gestire la situazione.
- La generalizzazione dell'entità NPC, di tipo totale ed esclusiva. Abbiamo scelto di far collassare verso il basso perché, avendo gli npc compiti diversi in base al sottotipo, sono legati ad entità diverse a seconda del ruolo che svolgono.
- La generalizzazione dell'entità Item, di tipo totale ed esclusiva. Abbiamo scelto di far collassare verso l'alto; questo perché i sottotipi presentano molti campi in comune tra loro e non svolgono azioni specifiche.
- La generalizzazione dell'entità Location, di tipo totale ed esclusivo. Come per l'entità NPC, abbiamo deciso di far collassare verso il basso; medesime sono anche le considerazioni fatte.


RPG/1
Account
AccountID
Username
Password
E-Mail
CreationDate
Premium
id: AccountID
Owns
0-4
1-1
0-80
Race
RaceID
RaceName
Description
id: RaceID
Belong
0-N
1-1
Character
CharacterID
CharacterName
ExpPoints
PlayerLevel
Health
Mana
Attack
Defense
Magic
Accuracy
Evasion
CriticalChance
id: CharacterID
IsA
0-N
Class
ClassID
ClassName
Main
Secondary
Description
id: ClassID
AvailableSkills
1-N
Skill
SkillID
SkillName
SkillLevel
Target
Effect
CastTime
ManaUsage
Cooldown
id: SkillID
LearnedSkills
QuestLog
QuestDate
StackOfItem
Number
Item
ItemID
ItemName
Type
Price
ItemLevel
Rarity
durability[0-1]
Subtype[0-1]
Damage[0-1]
DPS[0-1]
Defense[0-1]
Effect
CastTime[0-1]
id: ItemID
Ingredient
Amount
Recipe
RecipeID
id: RecipeID
CraftList
Crafter
CrafterID
CrafterName
id: CrafterID
Result
Merch
Trader
TraderID
TraderName
id: TraderID
Teaching
Trainer
TrainerID
TranerName
id: TrainerID
Fetch
Amount
Live_trad
Live_c
Live_trai
Live_v
Containment_v
Drop
DropPercentage
Reward
Village
VillageID
VillageName
id: VillageID
Villager
VillagerID
VillagerName
id: VillagerID
Proposition
EnemySkill
ChanceToUse
Field
FieldID
FieldName
FieldLevel
id: FieldID
Composition
Region
RegionID
RegionName
id: RegionID
Containment_d
Containment_a
Arena
ArenaID
ArenaName
Description
id: ArenaID
TakePlace
Dungeon
DungeonID
DungeonName
DungeonLevel
id: DungeonID
Population_f
Population_d
Winner
Loser
Fight
FightDate
id: Loser.Character
Loser.CharacterTakePlace.Arena
Loser.CharacterTakePlace.ArenaWinner.Character
FightDate
Enemy
EnemyID
EnemyName
Type
Classification
ExpProcured
EnemyLevel
Health
Mana
Attack
Defense
Temperance
id: EnemyID
Run
RunDate
Party
PartyID
PartyName
NumberOfMembers
id: PartyID
Complete
Partecipation
Target
Amount
Quest
QuestID
QuestName
ExpProcured
QuestLevel
isRepeatable
Type
id: QuestID

5.4 – PROGETTO LOGICO PER IL MODELLO RELAZIONALE

5.4.1 – Traduzione delle entità

La traduzione delle entità del nostro schema concettuale in relazioni è un processo a dir poco superfluo e che non richiede ulteriori commenti; occorre però far notare che queste non sono le relazioni definitive per il nostro progetto logico, in quanto la traduzione delle associazioni farà sì che, in taluni casi (associazioni uno a molti), queste relazioni vengano modificate tramite l'aggiunta di chiavi importate. Il campo sottolineato corrisponde alla chiave primaria.

- ACCOUNT:
  Account (AccountID, Username, Password, E-mail, CreationDate, Premium)
    AK: Username
    AK: E-mail

- ARENA:
  Arena (ArenaID, ArenaName, ArenaDescription)

- CHARACTER
  Character (CharacterID, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
    AK: CharacterName

- CLASS
  Class (ClassID, ClassName, Main, Secondary, Description)
    AK: ClassName

- CRAFTER
  Crafter (CrafterID, CrafterName)

- DUNGEON
  Dungeon (DungeonID, DungeonName, DungeonLevel)

- ENEMY
  Enemy (EnemyID, EnemyName, Type, Classification, ExpProcured, EnemyLevel, Health, Mana, Attack, Defense, Temperance)
  AK: EnemyName, EnemyLevel

- FIELD
  Field (FieldID, FieldName)

- FIGHT
  Fight (FightDate)

- ITEM
  Item (ItemID, ItemName, Type, Price, ItemLevel, Rarity, Durability, SubType, Damage, DPS, Defense, Effect, CastTime)

- PARTY
  Party (PartyID, PartyName, NumberOfMembers)
    AK: PartyName

- QUEST
  Quest (QuestID, Type, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)

- RACE
  Race (RaceID, RaceName, Description)
    AK: RaceName
- RECIPE
  Recipe (RecipeID)
- REGION
  Region (RegionID, RegionName)
- SKILL
  Skill (SkillID, SkillName, SkillLevel, Target, Effect, CastTime, ManaUsage, Cooldown)
    AK: SkillName, SkillLevel
- TRADER
  Trader (TraderID, TraderName)
- TRAINER
  Trainer (TrainerID, TrainerName)
- VILLAGE
  Village (VillageID, VillageName)
- VILLAGER
  Villager (VillagerID, VillagerName)

5.4.2 – Traduzione delle associazioni

La traduzione delle associazioni (che nel nostro schema concettuale sono tutte binarie, anche se di diverso tipo) modifica le relazioni che abbiamo appena introdotto per le entità. Infatti anche se le associazioni binarie di tipo molti a molti seguiranno la traduzione standard e implicheranno quindi la nascita di una nuova relazione avente come chiave le chiavi importate delle due entità che ne sono legate, le informazioni relative alle associazioni binarie uno a molti potranno essere inglobate all'interno di una delle relazioni già codificate per le entità legate dall'associazione, evitando l'introduzione di una nuova relazione. Esaminiamo la traduzione delle associazioni, passo dopo passo:

- AVAILABLE SKILL (lega CLASS e SKILL)
  Una classe ha disponibili più skill, allo stesso modo le skill possono essere disponibili per più di una classe. L'associazione è quindi di tipo m:n. Di conseguenza viene utilizzata la traduzione standard con tre relazioni; le entità Class e Skill rimangono invariate e viene introdotta la nuova entità AvailableSkill che ha come identificatore le chiavi importate dalle due entità partecipanti

  AvailableSkill (Class, Skill)
    FK: Class REFERENCES Class
    FK: Skill REFERENCES Skill

- CRAFT LIST (lega CRAFTER e RECIPE)
  Come nel caso precedente la relazione è di tipo m:n; anche la soluzione è quindi analoga

  CraftList (Crafter, Recipe)
    FK: Crafter REFERENCES Crafter
    FK: Recipe REFERENCES Recipe

- DROP (lega ENEMY e ITEM)
  La situazione è analoga a quelle precedenti, con l'unica differenza dell'introduzione di un campo DropPercentage proprio della nuova entità

  Drop (Enemy, Item, DropPercentage)
  FK: Enemy REFERENCES Enemy
  FK: Item REFERENCES Item

- ENEMY SKILL (lega ENEMY e SKILL)
  La situazione è analoga a quelle precedenti, con l'unica differenza dell'introduzione di un campo ChanceToUse proprio della nuova entità:

  EnemySkill (Enemy, Skill, ChanceToUse)
  FK: Enemy REFERENCES Enemy
  FK: Skill REFERENCES Skill

- INGREDIENT (lega RECIPE e ITEM)
  Come in EnemySkill la relazione presenta un proprio campo, stavolta nominato Amount:

  Ingredient (Recipe, Item, Amount)
  FK: Recipe REFERENCES Recipe
  FK: Item REFERENCES Item

- LEARNED SKILL (lega CHARACTER e SKILL)
  Analogo al caso di AvailableSkill

  LearnedSkill (Character, Skill)
  FK: Character REFERENCES Character
  FK: Skill REFERENCES Skill

- MERCH (lega TRADER e ITEM)
  Analogo al caso precedente

  Merch (Trader, Item)
  FK: Trader REFERENCES Trader
  FK: Item REFERENCES Item

- QUEST LOG (lega QUEST e CHARACTER)
  Come nel caso di EnemySkill la relazione ha un proprio campo, stavolta nominato QuestDate

  QuestLog (Quest, Character, QuestDate)
  FK: Quest REFERENCES Quest
  FK: Character REFERENCES Character

- RUN (lega PARTY e DUNGEON)
  Come nel caso precedente; il proprio campo è nominato RunDate

  Run (Party, Dungeon, RunDate)
  FK: Party REFERENCES Party
  FK: Dungeon REFERENCES Dungeon

- STACK OF ITEM (lega CHARACTER e ITEM)
  Come nel caso precedente; il proprio campo è nominato Number

  StackOfItem (Character, Item, Number)
  FK: Character REFERENCES Character
  FK: Item REFERENCES Item

- TEACHING (lega TRAINER e SKILL)
  Analogo al caso di AvailableSkill

  Teaching (Trainer, Skill)
  FK: Trainer REFERENCES Trainer
  FK: Skill REFERENCES Skill

- BELONG (lega CHARACTER e RACE)
  Dal momento che un character appartiene ad una e una sola race e una race può contare più di un character ci troviamo davanti ad una relazione 1:n; per tale motivo l'ID della race diventa un attributo della relazione character. La relazione character viene perciò modificata

  Character (CharacterID, Race, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
  FK: Race REFERENCES Race

- OWNS (lega ACCOUNT e CHARACTER)
  Anche questa associazione è di tipo 1:n, pertanto l'ID dell'account di appartenenza è presente nell'entità character.

  Character (CharacterID, Account, Race, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
  FK: Account REFERENCES Account

- IS A (lega CHARACTER e CLASS)
  Analoga alle precedenti; character mantiene un riferimento alla class

  Character (CharacterID, Class, Account, Race, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
  FK: Class REFERENCES Class

- RESULT (lega ITEM e RECIPE)
  Anche questa è una relazione 1:n, più precisamente un item può essere risultato di più recipe, mentre una recipe può produrre un solo item; di conseguenza verrà mantenuto nella relazione recipe l'ID dell'item prodotto

  Recipe (RecipeID, Result)
  FK: Result REFERENCES Item

- FETCH (lega QUEST e ITEM)
  Analoga alle precedenti; l'item può esser dovuto raccogliere in più quest, la quest fa raccogliere un solo tipo di item (anche se può variarne il numero da raccogliere). Se la quest non è di tipo FetchQuest questo campo sarà NULL.

Quest (QuestID, Item, Type, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)
FK: Item REFERENCES Item

- PROPOSITION (lega QUEST e VILLAGER)
  Una quest può essere proposta solo da un villager (anche più volte se la quest è ripetibile); lo stesso villager può proporre più quest

  Quest (QuestID, Villager, Item, Type, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)
  FK: Villager REFERENCES Villager

- TARGET (lega QUEST e ENEMY)
  Il ragionamento seguito è lo stesso dell'associazione Fetch, semplicemente sostituendo nel ragionamento l'enemy all'item. Se la quest non è di tipo KillQuest questo campo sarà NULL.

  Quest (QuestID, Enemy, Villager, Item, Type, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)
  FK: Enemy REFERENCES Enemy

- COMPLETE (lega QUEST e DUNGEON)
  Analogo al precedente, sostituendo dungeon ad enemy. Se la quest non è di tipo DungeonQuest questo campo sarà NULL.

  Quest (QuestID, Dungeon, Enemy, Villager, Item, Type, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)
  FK: Dungeon REFERENCES Dungeon

- PARTECIPATION (lega CHARACTER e PARTY)
  Un character può appartenere ad un solo party, un party può contenere fino a 5 membri. L'appartenenza di un character al party non è obbligatoria.

  Character (CharacterID, Party, Class, Account, Race, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
  FK: Party REFERENCES Party

- POPULATION_D (lega ENEMY e DUNGEON)
  Un nemico può popolare solo un dungeon, un dungeon può essere popolato da più enemy

  Enemy (EnemyID, Dungeon, EnemyName, Type, Classification, ExpProcured, EnemyLevel, Health, Mana, Attack, Defense, Temperance)
  FK: Dungeon REFERENCES Dungeon

- POPULATION_F (lega ENEMY e FIELD)
  Come sopra, sostituendo il field al dungeon

  Enemy (EnemyID, Field, Dungeon, EnemyName, Type, Classification, ExpProcured, EnemyLevel, Health, Mana, Attack, Defense, Temperance)
  FK: Field REFERENCES Field

- LIVE_TRAD (lega VILLAGE e TRADER)
  Un trader può vivere in un solo village; un village può essere casa di più trader (o anche di nessuno)

Trader (TraderID, Village, TraderName)
FK: Village REFERENCES Village

- LIVE_TRAI (lega TRAINER e VILLAGE)
  Come sopra, sostituendo trainer a trader

  Trainer (TrainerID, Village, TrainerName)
  FK: Village REFERENCES Village

- LIVE_V (lega VILLAGER e VILLAGE)
  Come sopra, sostituendo villager a trader

  Villager (VillagerID, Village, VillageName)
  FK: Village REFERENCES Village

- LIVE_C (lega CRAFTER e VILLAGE)
  Come sopra, sostituendo crafter a trader

  Crafter (CrafterID, Village, CrafterName)
  FK: Village REFERENCES Village

- WINNER (lega CHARACTER e FIGHT)
  Un character può vincere più fight; una fight può essere vinta da un solo character

  Fight (FightDate, Winner)
  FK: Winner REFERENCES Character

- LOSER (lega CHARACTER e FIGHT)
  Un character può perdere più fight; una fight può essere persa da un solo character

  Fight (FightDate, Loser, Winner)
  FK: Loser REFERENCES Character

- TAKE PLACE (lega FIGHT e ARENA)
  Un'arena può ospitare più fight; una fight può essere ospitata da una sola arena

  Fight (FightDate, Arena, Loser, Winner)
  FK: Arena REFERENCES Arena

- CONTAINMENT_V (lega FIELD e VILLAGE)
  Un villaggio è contenuto in un solo field; un field può contenere più village (o anche nessuno)

  Village (VillageID, Field, VillageName)
  FK: Field REFERENCES Field

- CONTAINMENT_A (lega FIELD e ARENA)
  Come sopra, sostituendo village con arena

  Arena (ArenaID, Field, ArenaName)
  FK: Field REFERENCES Field

- CONTAINMENT_D (lega FIELD e DUNGEON)
  Come sopra, sostituendo village con arena

Dungeon (DungeonID, Field, DungeonName)
FK: Field REFERENCES Field

- COMPOSITION (lega FIELD e REGION)
  Una regione è composta di field; un field fa parte di una sola regione

  Field (FieldID, Region, FieldName)
  FK: Region REFERENCES Region

5.5 - Traduzione delle operazioni in Query SQL

In questa sezione tradurremo tutte le singole operazioni elementari che abbiamo finora analizzato solo attraverso gli schemi di navigazione dell'E/R in vere e proprie query in linguaggio SQL.

5.6.1 Aggiunta di un nuovo Dungeon

INSERT INTO Dungeon (Field, DungeonName, Level)

VALUES (*FieldDesiderato, DungeonNameDesiderato, LevelDesiderato*)

5.6.2.1 Visualizzazione del livello medio per ciascuna classe

SELECT cl.ClassName, AVG(ch.PlayerLevel) AS LVLMedio

FROM Character as ch, Class as cl

WHERE ch.Class = cl.ClassID and ch.PlayerLevel > 0

GROUP BY cl.ClassName

ORDER BY AVG(ch.PlayerLevel) DESC

5.6.2.2 Visualizzazione delle classi più vincenti nei duelli

SELECT cl.ClassName, COUNT(f.Winner) as Victories

FROM Class as cl, Character as ch, Fight as f

WHERE f.Winner = ch.CharacterID and ch.Class = cl.ClassID

GROUP BY cl.ClassName

ORDER BY COUNT(f.Winner) DESC

5.6.3.1 Selezione dell'arena specificata

SELECT ArenaName

FROM Arena

5.6.3.2 Visualizzazione dei giocatori con più vittorie in tale arena

SELECT TOP 2 a.ArenaName, c.CharacterName, COUNT(*) as Victories

FROM Arena as a, Fight as f, Character as c

WHERE ((a.ArenaName = "*arenaName*")

AND (a.ArenaID = f.Arena)

AND (c.CharacterID = f.Winner))

GROUP BY a.ArenaName, c.CharacterName, f.Winner

ORDER BY a.ArenaName, COUNT(*) DESC

5.6.4 Visualizzazione degli ultimi personaggi con cui si è combattuto

SELECT CharacterID, CharacterName, COUNT(*) as NumberOfFights

FROM (**SELECT** TOP 5 c.CharacterID, c.CharacterName

FROM Fight AS f, Character AS c

WHERE (((f.Winner) = *characterID*)

AND ((f.Loser) = c.CharacterID))

OR (((f.Winner) = c.CharacterID)

AND ((f.Loser) = *characterID*))

ORDER BY f.FightDate DESC)

GROUP BY CharacterID, CharacterName

5.6.5.1 Selezione dell'abilità che si vuole imparare

SELECT SkillName

FROM Skill

5.6.5.2 Visualizzazione dei luoghi in cui è possibile impararla

SELECT tr.TrainerName, v.VillageName, f.FieldName, r.RegionName

FROM Region AS r, Field AS f, Village AS v, Skill AS s, Trainer AS tr, Teaching AS te

WHERE (((r.RegionID) = f.Region)

AND ((s.SkillID) = te.Skill)

AND ((tr.TrainerID) = te.Trainer)

AND ((v.VillageID) = tr.Village)

AND ((f.FieldID) = v.Field)

AND ((s.SkillName) = "*skillName*"))

ORDER BY r.RegionID

5.6.6.1 Selezione dell'oggetto da ispezionare

SELECT ItemName

FROM Item

5.6.6.2 Visualizzazione dei modi in cui si può ottenere

SELECT t.TraderName AS Name, "Trader" AS Type

FROM Trader t, Merch m

WHERE t.TraderID = m.Trader

AND m.Item = *itemID*

UNION SELECT e.EnemyName AS Name, "Enemy Drop" AS Type

FROM Enemy e, Drop d

WHERE e.EnemyID = d.Enemy

AND d.Item = *itemID*

UNION SELECT c.CrafterName AS Name, "Crafter" AS Type

FROM Crafter c, CraftList l, Recipe r

WHERE c.CrafterID = l.Crafter

AND r.RecipeID = l.Recipe

AND r.RecipeID = *itemID*

UNION SELECT q.QuestName AS Name, "Quest" AS Type

FROM Quest q

WHERE q.Reward = *itemID*

5.6.7.1 Selezione del villaggio

SELECT VillageName

FROM Village

5.6.7.2 Visualizzazione degli NPC del villaggio scelto

SELECT c.CrafterName AS Name

FROM Crafter c, Village v

WHERE v.VillageName = "*villageName*"

AND c.Village=v.VillageID

UNION SELECT t.TraderName

FROM Trader t, Village v

WHERE v.VillageName = "*villageName*"

AND t.Village = v.VillageID

UNION SELECT t.TrainerName

FROM Trainer t, Village v

WHERE v.VillageName = "*villageName*"

AND t.Village = v.VillageID

UNION SELECT vi.VillagerName

FROM Villager vi, Village v

WHERE v.VillageName = "*villageName*"

AND vi.Village = v.VillageID

6 – IL PROGETTO FISICO

6.1 – Indicizzazione atrributi

Questa sezione si occupa di modellare ulteriormente il progetto logico secondo raffinamenti che prendono in considerazione l'organizzazione fisica dei dati. Illustreremo in particolare tutti gli indici previsti sugli attributi delle relazioni individuate nel corso della sezione precedente, opportunamente inseriti allo scopo di ridurre i tempi di ricerca e i costi di accesso alle relazioni stesse. Segue l'elenco delle relazioni sulle quali abbiamo previsto degli indici:

- **Account** (AccountID, Username, Password, E-Mail, CreationDate, Premium)
  Indice su AccountID perché chiave primaria

- **Arena** (ArenaID, Field, ArenaName, Description)
  Indice su ArenatID perché chiave primaria

- **AvailableSkill** (Class, Skill)
  Indice su Class e Skill perché chiave primaria

- **Character** (CharacterID, Account, Class, Race, Party, CharacterName, ExpPoints, PlayerLevel, Health, Mana, Attack, Defense, Magic, Accuracy, Evasion, CriticalChance)
  Indice su CharacterID perché chiave primaria

- **Class** (ClassID, ClassName, Main, Secondary, Description)
  Indice su ClassID perché chiave primaria

- **Crafter** (CrafterID, Village, CrafterName)
  Indice su CrafterID perché chiave primaria e su Village perché spesso usato in query di selezione come attributo di join

- **CraftList** (Crafter, Recipe)
  Indice su Crafter e Recipe perché chiave primaria

- **Drop** (Enemy, Item, DropPercentage)
  Indice su Enemy e Item perché chiave primaria

- **Dungeon** (DungeonID, Field, DungeonName, DungeonLevel)
  Indice su DungeonID perché chiave primaria

- **Enemy** (EnemyID, Field, Dungeon, EnemyName, Type, Classification, ExpProcured, EnemyLevel, Health, Mana, Attack, Defense, Temperance)
  Indice su EnemyID perché chiave primaria

- **EnemySkill** (Enemy, Skill, ChanceToUse)
  Indice su Enemy e Skill perché chiave primaria

- **Field** (FieldID, Region, FieldName, FieldLevel)
  Indice su FieldID perché chiave primaria e su Region perché a volte viene usato in query di selezione come attributo di join

- **Fight** (FightDate, Arena, Winner, Loser)
  Indice su FightDate, Arena, Winner e Loser perché chiave primaria

- **Ingredient** (Recipe, Item, Amount)
  Indice su Recipe e Item perché chiave primaria

- **Item** (ItemID, ItemName, Type, Price, ItemLevel, Rarity, Durability, SubType, Damage, DPS, Defense, Effect, CastTime)
  Indice su ItemID perché chiave primaria

- **LearnedSkill** (Character, Skill)
  Indice su Character e Skill perché chiave primaria

- **Merch** (Trader, Item)
  Indice su Trader e Item perché chiave primaria

- **Party** (PartyID, PartyName, NumberOfMembers)
  Indice su PartyID perché chiave primaria

- **Quest** (QuestID, Villager, Reward, Type, Dungeon, Enemy, Item, Amount, QuestName, ExpProcured, QuestLevel, IsRepeatable)
  Indice su QuestID perché chiave primaria

- **QuestLog** (Character, Quest, QuestDate)
  Indice su Character, Quest e QuestDate perché chiave primaria

- **Race** (RaceID, RaceName, Description)
  Indice su RaceID perché chiave primaria

- **Recipe** (RecipeID, Result)
  Indice su RecipeID perché chiave primaria

- **Region** (RegionID, RegionName)
  Indice su RegionID perché chiave primaria

- **Run** (Party, Dungeon, RunDate)
  Indice su Party, Dungeon e RunDate perché chiave primaria

- **Skill** (SkillID, SkillName, SkillLevel, Target, Effect, CastTime, ManaUsage, Cooldown)
  Indice su SkillID perché chiave primaria

- **StackOfItem** (Character, Item, Number)
  Indice su Character e Item perché chiave primaria

- **Teaching** (Trainer, Skill)
  Indice su Trainer e Skill perché chiave primaria

- **Trader** (TraderID, Village, TraderName)
  Indice su TraderID perché chiave primaria e su Village perché spesso usato in query di selezione come attributo di join

- **Trainer** (TrainerID, Village, TrainerName)
  Indice su TrainerID perché chiave primaria e su Village perché spesso usato in query di selezione come attributo di join

- **Village** (VillageID, Field, VillageName)
  Indice su VillageID perché chiave primaria e su Field perché a volte viene usato in query di selezione come attributo di join

- **Villager** (VillagerID, Village, VillagerName)
  Indice su VillagerID perché chiave primaria e su Village perché spesso usato in query di selezione come attributo di join

6.2 – Ordinamento su attributi

- **Arena** (ArenaID, Field, ArenaName, Description)
  Ordinamento sul campo ArenaName perché richiesto in uscita e per facilitare l'esecuzione di alcune query

- **Class** (ClassID, ClassName, Main, Secondary, Description)
  Ordinamento sul campo ClassName perché richiesto in uscita e per facilitare l'esecuzione di alcune query

7 – L'Interfaccia utente

L'interfaccia utente del nostro elaborato è un prototipo di ciò che dovrà essere l'interfaccia utente del gioco effettivo. L'applicazione è stata realizzata in linguaggio Java, utilizzando la libreria grafica Swing e UCanAccess per interfacciarsi al database. Essa si deve occupare di gestire due diversi tipi di utenti: gli amministratori e i giocatori; perciò nella schermata iniziale sarà possibile scegliere la propria categoria, ottenendo l'accesso a servizi esclusivi. Passiamo ora ad esaminare i due tipi di interfacce e le diverse funzioni da esse offerte.

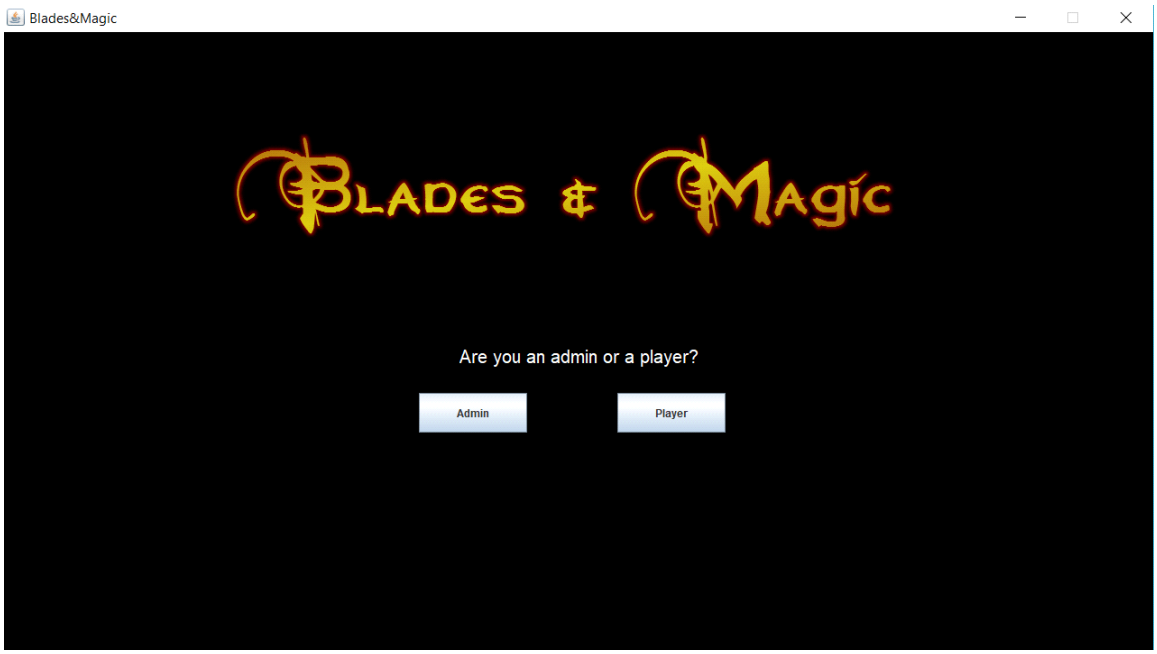


Figura 7.1 - La schermata iniziale

7.1 - L'Amministratore

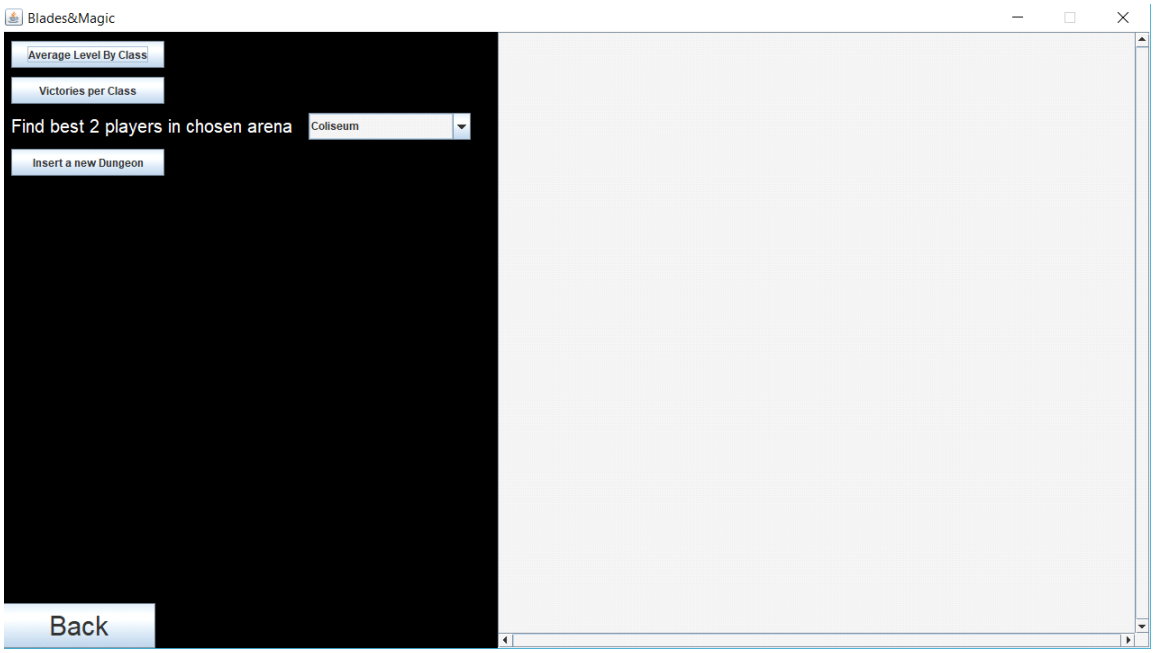


Figura 7.2 - La schermata dell'amministratore

Una volta effettuato l'accesso, l'applicativo offre all'amministratore le seguenti possibilità:

- Average Level By Class
- Victories per Class
- Find best two players in chosen arena
- Insert a new Dungeon
- Back

Il tasto "Average Level By Class" permette di visualizzare il livello medio dei personaggi di ciascuna classe; il tasto "Victories per Class" permette di visualizzare quante vittorie hanno accumulato complessivamente i personaggi di ciascuna classe; il menu a tendina sottostante permette di scegliere un'arena e visualizzare i migliori due giocatori che ci hanno combattuto; il tasto "Insert a new Dungeon" permette di inserire un nuovo Dungeon; infine, il tasto "Back" permette di tornare alla schermata iniziale.
Nella parte destra della schermata verranno visualizzati i risultati delle richieste sopra elencate sotto forma di tabella.

7.2 - Il Giocatore

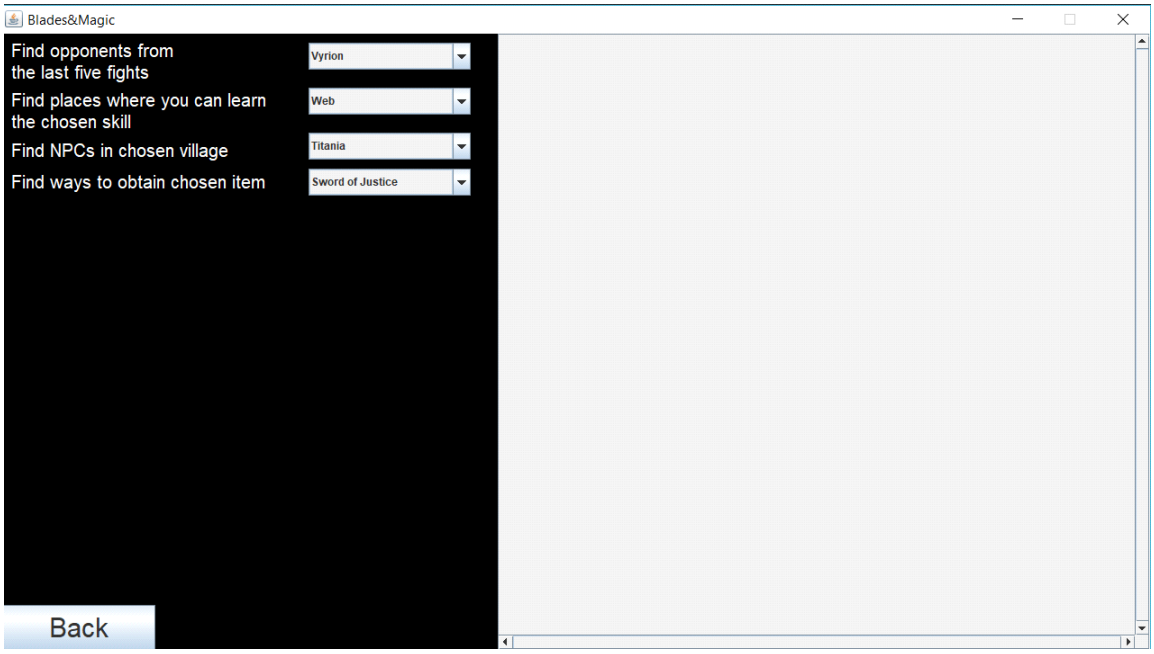


Figura 7.3 - La schermata del giocatore

Una volta effettuato l'accesso, l'applicativo offre al giocatore le seguenti possibilità:

- Find opponents from the last five fights
- Find places where you can learn the chosen skill
- Find NPCs in chosen village
- Find ways to obtain chosen item
- Back

Il primo menu a tendina consente di visualizzare gli ultimi giocatori sfidati nelle ultime cinque battaglie (nell'ambito di questa applicazione si è deciso di scegliere il personaggio desiderato attraverso il menù a tendina; nel gioco effettivo la scelta ricadrebbe sul personaggio con il quale si ha effettuato il login); il secondo menu a tendina permette di visualizzare i luoghi in cui imparare la skill selezionata e chi può insegnarcela; il terzo consente di visualizzare gli NPC presenti in un determinato villaggio; il quarto permette di visualizzare i modi di ottenere un determinato oggetto; il tasto "Back" ha la stessa funzione descritta nel paragrafo precedente. Come per l'amministratore, la parte destra dello schermo viene utilizzata per mostrare i risultati delle richieste di cui sopra.

8 - Glossario

TERMINE	SIGNIFICATO
Boss	In genere, il boss è l'ultimo nemico che si affronta in un dungeon ed è anche il più impegnativo da sconfiggere.
Character	Sinonimo di Personaggio. Si veda la voce "personaggio"
Class	Questo termine indica la classe del personaggio, che in genere determina il suo stile di gioco. Ad esempio un Assassino predilige la mobilità e i danni elevati, un Mago si specializza nei colpi magici, un Tiratore combatte dalla distanza ma è debole nel corpo a corpo, ecc.
Consumable	Categoria di oggetto che accomuna tutti quegli oggetti che si consumano una volta attivatone l'effetto. Un esempio sono gli elisir e le pozioni.
Craft, Crafting	Craft significa letteralmente "costruire". Spesso nei giochi di ruolo si implementano sistemi di crafting nel senso che si dà la possibilità ai giocatori di costruire oggetti a partire dalle materie prime.
Drop, Droppare	Con drop si indica l'atto di lasciare al suolo degli oggetti quando si muore. Spesso si usa il verbo italianizzato "droppare" per indicare ciò. Si usa quasi sempre quando un nemico viene sconfitto e lascia a terra degli oggetti. Per esempio un nemico come un drago potrebbe droppare scaglie di drago, zanne ecc.
Dungeon	Dungeon significa prigione, oppure segrete, oppure sotterranei. Nei Giochi di Ruolo con questo termine si indica un percorso popolato da nemici che va affrontato per ottenere una ricompensa finale (ad esempio, un dungeon potrebbe essere un covo di banditi, una caverna con un drago che custodisce un tesoro, i sotterranei di un castello dove è stato rinchiuso un mostro…)
Exp, Experience	Un elemento ricorrente nei giochi di ruolo sono i Punti Esperienza. I punti esperienza sono un punteggio assegnato al giocatore quando sconfigge nemici o completa missioni. Quando i punti esperienza superano una certa soglia, il giocatore "sale di livello" ovvero aumenta le proprie statistiche e può imparare nuove abilità. In alcuni giochi si può salire di livello all'infinito, mentre in altri c'è un limite di livello massimo.
Fetch, Fetch Quest	Fetch significa letteralmente "andare a prendere". Le Quest di tipo fetch infatti consistono nel collezionare un certo numero di oggetti. Una fetch Quest ad esempio potrebbe consistere nel raccogliere 20 gusci di conchiglia per preparare una medicina per un NPC che sta male
Field	Un elemento ricorrente dei giochi di ruolo sono i Field. Nella mappa, i Field sono campi aperti costellati di luoghi di interesse che è possibile visitare. Ad esempio, un Field potrebbe essere una foresta che contiene il villaggio degli elfi, la caverna dei goblin, la sorgente della giovinezza e così via.
Inventory	L'inventario di un personaggio è l'insieme degli oggetti (armi, armature, pozioni, tesori, oggetti di vario genere) che si porta con sé.
MMORPG	Massive Multiplayer Role Playing Game, gioco di ruolo online progettato per essere giocato contemporaneamente da un elevato numero di giocatori
NPC	Non-Playable Character. Con questo termine si indicano i personaggi gestiti dal computer. Quasi sempre con questo termine si indicano tutti quei personaggi non ostili con cui si può interagire, ad esempio conversando o compravendendo oggetti.
Party	I party sono un gruppo di personaggi che collaborano per completare missioni e dungeon. Spesso i Party sono formati da personaggi di classi differenti, in maniera tale che ogni personaggio possa bilanciare i punti di debolezza degli altri (ad esempio i tiratori sono forti dalla distanza ma hanno poca difesa, i guerrieri invece combattono corpo a corpo e permettono di tenere i nemici lontani dagli elementi

	più fragili del gruppo)
Personaggio	In genere nei giochi di ruolo si vestono i panni di un proprio personaggio personalizzato, che è protagonista della storia.
PVP	Acronimo di Player Versus Player, sigla indicata per descrivere le attività dove si sfidano tra loro più giocatori umani.
Quest	Una quest è una missione affidata al giocatore. Quasi sempre le quest offrono come ricompense denaro, beni materiali o punti esperienza. A seconda del gioco le quest vengono affidate al giocatore dagli NPC oppure in maniera automatica.
Quest Log	Un Quest Log è un elemento ricorrente dei Giochi di Ruolo che consiste in una sorta di diario personale che tiene traccia delle missioni completate.
Race	Razza. In genere i personaggi dei giocatori, nei giochi di ruolo, appartengono a varie razze diverse (Ad esempio umani, elfi, nani…).
Region	Un elemento ricorrente del mondo dei Giochi di Ruolo sono le regioni. Le regioni sono comparabili ad un continente, o a un regno, e in genere si compongono di tanti Field.
RPG	Sinonimo di Gioco di Ruolo (GdR). Genere di gioco dove il giocatore veste i panni di un personaggio. Il gioco può svilupparsi in tanti modi diversi ma in genere il personaggio del giocatore affronta dei nemici, guadagna esperienze dalla battaglia e diventa più forte.
Run	L'atto di affrontare un Dungeon da parte di uno o più giocatori insieme viene detto Run.
Skill	Per skill si intende un'abilità speciale, che sia essa appartenente a un nemico o al giocatore. Ad esempio, una skill potrebbe essere "Fireball" che lancia una palla di fuoco, o "Protection" che protegge dai danni. In generale non si fa molta distinzione tra magie e altri tipi di mosse, col termine Skill si indica tutto quanto.
Stack, Stack of Items	Stack significa letteralmente "pila". Spesso nei giochi di ruolo, gli oggetti del giocatore nell'inventario vengono organizzati in pile, nel senso che più copie di uno stesso oggetto vengono memorizzate nello stesso slot, in cui si indica anche la quantità di tale oggetto.