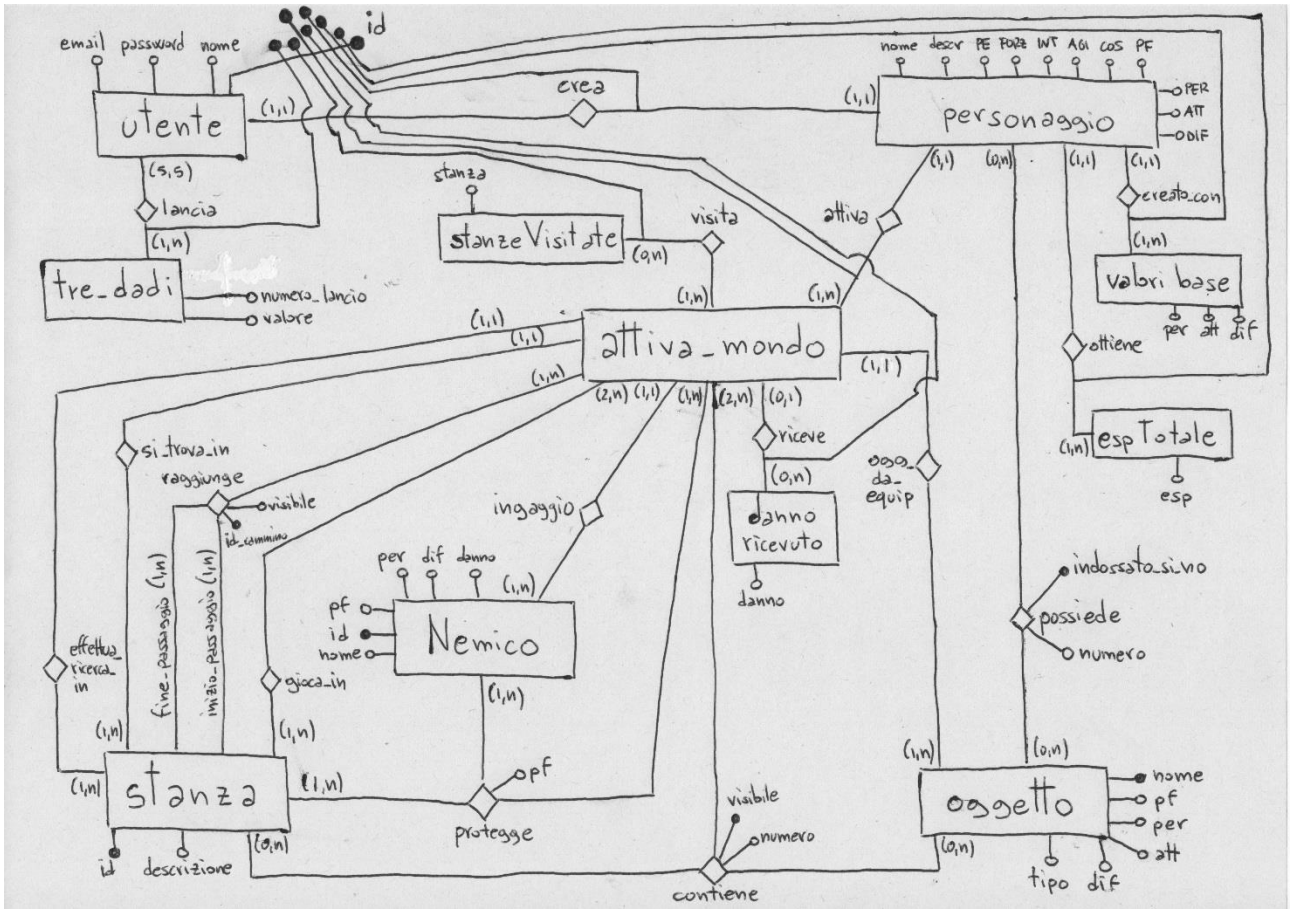
Progetto di Basi di Dati

DOCUMENTAZIONE TECNICA

Gabriele Cazzaniga Matricola: 859738 e-mail:
gabriele.cazzaniga@studenti.unimi.it

SCHEMA CONCETTUALE (ER)



L'applicazione di basi di dati è volta a gestire la creazione di un personaggio, l'aggiornamento del mondo di gioco a seguito di un combattimento, l'equipaggiamento dei personaggi e verificare i vincoli relativi al possesso di oggetti, infine generare il dungeon all'inizio dell'avventura.
Si possono quindi individuare delle entità fondamentali per lo sviluppo di queste funzionalità:

- UTENTE, TRE_DADI, PERSONAGGIO, VALORI BASE e ESP_TOTALE
- NEMICO
- OGGETTO
- STANZA
- DANNO RICEVUTO e STANZE VISITATE
- ATTIVA_MONDO

UTENTE, TRE_DADI, PERONAGGIO, VALORI BASE, ESP TOTALE

Si occupano della creazione del personaggio, come si può notare vengono tutte identificate dalla chiave primaria dell'UTENTE, in quanto è stato deciso che un utente può possedere un unico personaggio.

Al momento della registrazione dell'utente verrà automaticamente creato un personaggio di default dalla funzione pers_def().

L'idea è fondamentalmente stata quella di separare la gestione del personaggio e la creazione del mondo per il personaggio creato. Verrà quindi utilizzata l'entità ATTIVA_MONDO, contenente solamente l'identificativo del personaggio, per gestire tutto l'ambiente di gioco del personaggio creato.

Il vantaggio di quest'entità intermedia tra il personaggio e il mondo in cui questo gioca è che per fare in modo che il mondo generato sia ogni volta diverso basta eliminare e reinserire l'ID del personaggio in ATTIVA_MONDO. Così facendo, automaticamente verranno cancellate tutte le informazioni relative al vecchio mondo e verranno inserite quelle nuove senza modificare quelle qualità che il personaggio si è voluto mantenesse, quali i valori di base di creazione (VALORI BASE) e l'esperienza totale accumulata (ESP TOTALE).

NEMICO

In quest'entità vengono inseriti staticamente tutti i nemici possibili che l'utente dovrà affrontare con le rispettive statistiche. Questi non verranno mai modificati.

OGGETTO

In quest'entità vengono inseriti staticamente tutti gli oggetti possibili che l'utente potrà utilizzare con le rispettive statistiche. Questi non verranno mai modificati.

STANZA

In quest'entità vengono inserite staticamente tutte le stanze possibili che l'utente potrà visitare con la rispettiva descrizione. Queste non verranno mai modificate.

DANNO RICEVUTO, STANZE VISITATE

Queste due entità si occupano rispettivamente di contenere il danno che il nemico è riuscito ad infliggere al personaggio e le stanze che il personaggio ha visitato durante il gioco.

Si è pensato di creare queste entità appositamente per calcolare l'esperienza che il personaggio ottiene passo passo durante il gioco, alla fine se il personaggio riuscirà a terminare vittorioso il gioco, la sua PE verrà aggiunta alla sue ESP TOTALE, se no verrà semplicemente resettata.

ATTIVA_MONDO

Quest'entità è quella fondamentale e di riferimento, ha il compito di creare il mondo di gioco (ogni volta diverso) per il personaggio attraverso l'attivazione della funzione crea_mondo() e collegando tutte le entità necessarie: PERSONAGGIO,NEMICO,OGGETTO e STANZA.

Per fare ciò utilizza diverse funzioni, concettualmente collega le entità interessate con queste relazioni:

- decidere quante stanze saranno presenti nel mondo (gioca_in)
- decidere i collegamenti tra le stanze (raggiunge)
- decidere quanti e quali nemici saranno presenti nelle stanze (protegge)
- decidere quanti e quali oggetti saranno presenti nelle stanze (contiene)

INFORMAZIONI AGGIUNTIVE

- la relazione EFFETTUA_RICERCA_IN risulta molto comoda per l'attivazione del trigger che richiama la procedura per ricercare oggetti all'interno della stanza specificata.

- La relazione SI_TROVA_IN risulta molto comoda per l'attivazione del trigger che richiama la procedura per gestire gli oggetti che il personaggio trova o potrebbe trovare nella stanza specificata.

- La relazione OGG_DA_EQUIP risulta molto comoda per l'attivazione del trigger che richiama la procedura per equipaggiare l'oggetto specificato tenendo conto dei vincoli imposti.

SCHEMA RELAZIONALE

UTENTE (id, email, password, nome)

TRE_DADI (id, numero_lancio, valore)

PERSONAGGIO (id, nome, descrizione, PE, FORZ, INT, AGI, COS, PF, PER, ATT, DIF)

OGGETTO (nome, pf, per, att, dif, tipo)

POSSIEDE (id_personaggio, nome_oggetto, indossato_si_no, numero)

STANZA (id, descrizione)

GIOCA_IN (id_personaggio, id_stanza)

ATTIVA_MONDO (id_personaggio)

NEMICO (id, nome, descrizione, pf, per, dif, danno)

PROTEGGE (id_personaggio, id_stanza, id_nemico, pf)

RAGGIUNGE (id_cammino, id_personaggio, inizio_passaggio, fine_passaggio, visibile)

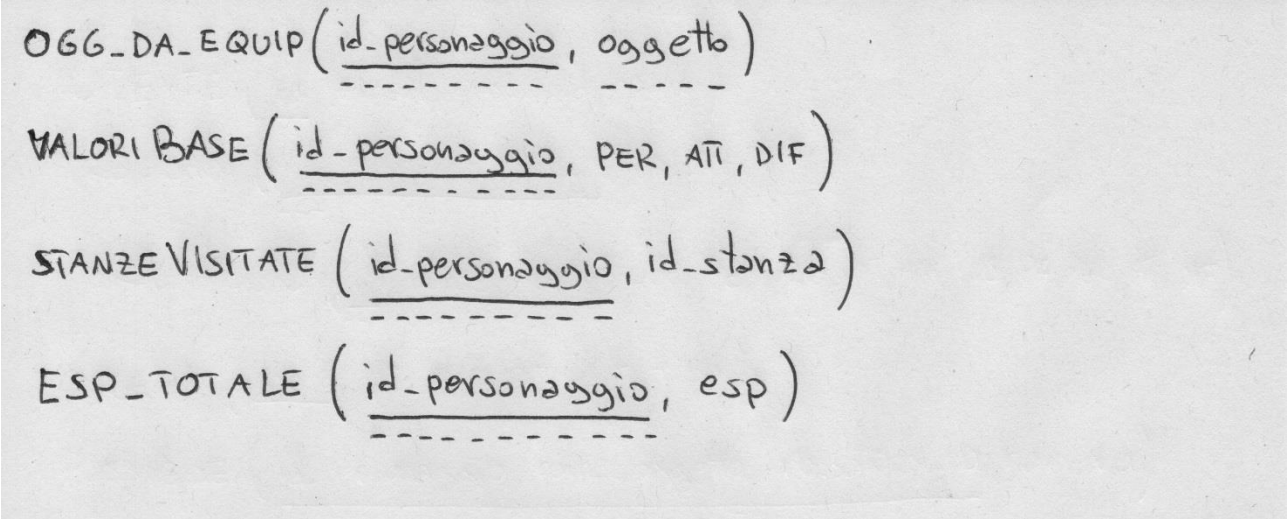
CONTIENE (id_personaggio, id_stanza, nome_oggetto, visibile, numero)

INGAGGIO (id_personaggio, nemico_ingaggiato)

DANNO_RICEVUTO (id_personaggio, danno)

SI_TROVA_IN (id_personaggio, stanza)

EFFETTUA_RICERCA_IN (id_personaggio, stanza)



DETTAGLI IMPLEMENTATIVI

Prodotti software utilizzati

- Database: *PostgreSQL*
- Server web: *Apache HTTP server*
- Visualizzazione delle pagine web: web browser come *Google Chrome*

Linguaggi utilizzati

- Creazione e interrogazione della base di dati: *SQL* (di PostgreSQL)
- Procedure interne alla base di dati: *PLPGSQL*
- Creazione di pagine web: *HTML5*
- Decorazione delle pagine web: *CSS*
- Questioni da gestire dal lato server: *PHP*
- Aspetti da elaborare dal lato client: *JavaScript*

PROCEDURE REALIZZATE

elimina_cons():

questa funzione si occupa di eliminare gli oggetti consumabili equipaggiati ogni volta che viene cambiata stanza. Decrementa o aumenta le caratteristiche del personaggio a seconda dei bonus che gli erano stati assegnati dagli oggetti che aveva consumato nella stanza precedente, dopodiché elimina gli oggetti consumati.

pers_def():

questa funzione si occupa di creare un personaggio di default associato all'utente appena registrato, inserisce quindi nelle tabelle interessate valori di default che poi verranno modificati con la creazione passo passo del personaggio da parte dell'utente.

equipaggia():

questa funzione si occupa di equipaggiare gli oggetti selezionati dall'utente al personaggio. E' stato deciso di considerare come oggetti il fatto di non avere equipaggiata alcun arma e alcun armatura, il personaggio così risulta equipaggiato di default con Nessun Arma e Nessun Armatura.
La funzione tiene conto del fatto che solo un oggetto di tipo attacco o difesa può essere equipaggiato, se quindi l'utente decide di equipaggiare un altro oggetto di attacco o difesa verrà disequipaggiato quello precedentemente equipaggiato.
Siccome si è scelto che il personaggio può equipaggiare più oggetti di tipo consumabile nello stesso momento, e siccome si è scelto che il personaggio può possedere più oggetti consumabili dello stesso tipo (es. 2 razioni di cibo), la funzione aumenta il numero di oggetti equipaggiati a seconda di quanti di questi sono in quel momento equipaggiati e decrementa quindi il numero di oggetti ancora da equipaggiare di quel tipo.

infliggi():

questa funzione si occupa semplicemente di aggiornare la vita del personaggio se l'attacco di un nemico va a buon fine.
Prende quindi il valore del danno dal nemico che ha colpito il personaggio e lo sottrae alla vita del personaggio.

aggiorna():

questa funzione si occupa di aggiornare le caratteristiche del personaggio a seconda degli oggetti che in quel momento ha equipaggiato.
Prende quindi le caratteristiche degli oggetti equipaggiati e le somma/sottrare ai valori del personaggio.

combatti():

questa funzione si occupa di aggiornare la vita del nemico se l'attacco del personaggio va a buon fine. Prende quindi il valore dei punti ferita dell'arma equipaggiata al personaggio in quel momento e li sottrae ai pf del nemico.
Se i pf del nemico arrivano a zero, il nemico viene eliminato dalla tabella che associa ogni nemico alla protezione di una stanza.
Contemporaneamente a questo, questa funzione si occupa anche di registrare l'esperienza che il personaggio ottiene eliminando un nemico, inserisce quindi il valore del danno del nemico nei PE del personaggio che l'ha eliminato.

get_random_number(INTEGER, INTEGER):

questa funzione genera un numero pseudo-casuale tra i due valori forniti in ingresso.

crea_Mondo():

questa funzione si occupa di creare il mondo in modo casuale, deve quindi gestire tutti gli aspetti visti precedentemente con l'analisi dell'entità ATTIVA_MONDO.

- vengono scelti 6 numeri casuali compresi tra intervalli ben definiti, che verranno usati per:
    - Decidere il numero di stanze che saranno contenute nel mondo
    - Abbinare i nemici che verranno inseriti nelle stanze
        - I nemici sono ordinati per forza con identificativi in ordine crescente, verranno quindi selezionati per ogni stanza considerano i numeri casuali estratti
            - o 3 nemici di basso livello
            - o 2 nemici uno di basso livello e uno di alto livello
            - o 1 nemico di livello medio
        - Ci sono due nemici speciali chiamati Taran e Junior che in mutua esclusione possono essere inseriti in una stanza
            - Se viene inserito Taran, in quanto molto potente, gli altri avversari, dopo il suo inserimento, saranno selezionati tra i più deboli
            - Se viene inserito Junior, in quanto molto scarso, gli altri avversari, dopo il suo inserimento, saranno selezionati tra i più forti
- Vengono resettati i pe del personaggio, così che il personaggio sia pronto a ricominciare la raccolta di esperienza da zero

- Viene scelto un numero casuale che verrà utilizzato per determinare il numero di passaggi presenti in una stanza e un altro numero casuale per stabilire se sarà visibile oppure no
- Vengono poi inserite la stanza iniziale e quella finale nella tabella che contiene le stanze in cui il personaggio andrà a giocare, in quanto deve sempre esserci una stanza iniziale ed una finale
- Per creare più strade che portano alla stanza finale vengono poi inseriti 3 passaggi visibili/no raggiungibili da stanze scelte in modo casuale
- Vengono quindi inseriti tutti gli oggetti di attacco e di difesa, sempre in stanze casuali e con visibilità casuale.
- Dopodiché a seconda di quante stanze sono state riservate al mondo creato verranno inseriti un diverso numero di oggetti consumabili, anch'essi sempre in modo casuale con visibilità casuale. Siccome gli oggetti sono anche caratterizzati da un numero, viene effettuato anche un controllo sul numero di oggetti già presenti visibili/no e nel caso aggiornato
- Infine vengono inseriti dei valori di default nelle tabelle da aggiornare successivamente nel caso debbano accadere eventi (come ad esempio l'equipaggiamento di un arma)

raccogli():

questa funzione si occupa di fare in modo che quando un personaggio entra in una stanza, tutti gli oggetti visibili vengano raccolti automaticamente rispettando i vincoli imposti. Viene quindi sempre gestita l'eccezione degli oggetti consumabili che sono caratterizzati dal numero di oggetti dello stesso tipo consumabili già posseduti equipaggiati o no.

ricerca():

questa funzione si occupa della ricerca di oggetti nascosti o passaggi nascosti nella stanza selezionata. Deve sempre tener conto del fatto che il personaggio potrebbe già possedere un oggetto di tipo consumabile dello stesso tipo e quindi aumentarne il numero. Deve tenere anche conto del fatto che se è stato raggiunto il massimo numero di oggetti posseduti, l'oggetto non deve poter essere recuperato dal personaggio. Infine deve selezionare un oggetto nascosto presente nella stanza oppure una stanza nascosta in modo casuale.
Se è stata selezionata una stanza basterà quindi renderla visibile,
se è stato selezionato un oggetto, questo sarà inserito negli oggetti in possesso del personaggio e poi verrà eliminato dalla stanza.