Dungeons as DB

Andrei Ciulpan

Basi di Dati 2016-2017

Funzioni implementate nel DB

Note: Tutte le funzioni (29) sono state scritte nel linguaggio procedurale PLpg/SQL

Triggers

- ```
  CREATE TRIGGER delbonusoggetti_trigger
  BEFORE DELETE ON dungeonsasdb.equipaggia
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.delbonusoggetti()
  ```

  Agisce prima di DELETE su Equipaggia, ovvero quando il personaggio disequipaggia un oggetto:

  ```
  -- rimuove i bonus dell'oggetto disequipaggiato dal personaggio
  -- update degli attributi Arma_EQ e Armatura_EQ
  ```

- ```
  CREATE TRIGGER gestioneequipaggia_trigger
  AFTER INSERT OR UPDATE ON dungeonsasdb.equipaggia
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.gestioneequipaggia()
  ```

  Agisce dopo INSERT o UPDATE su Equipaggia, ovvero quando il personaggio equipaggia un oggetto:

  ```
  -- aggiunge i bonus dell'oggetto equipaggiato al personaggio
  -- provvede che il personaggio non può equipaggiare piu' di
     1 arma, 1 armatura oppure 2 gioielli
  -- update degli attributi Arma_EQ, Armatura_EQ
  -- cancella da Possiede gli oggetti di tipo pozione e razione dopo
     l'utilizzo (ovvero dopo l'equipaggiamento)
  ```

- ```
  CREATE TRIGGER aggiornastanzapassaggi_trigger
  AFTER INSERT OR DELETE ON dungeonsasdb.passaggio
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.aggiornastanzapassaggi()
  ```

  Agisce dopo INSERT o DELETE su Passaggio:
  ```

```
        -- Se INSERT: aumenta di 1 il valore di NR_PASS su Stanza
        -- Se DELETE: diminuisce di 1 il valore di NR_PASS su Stanza
```

- ```
  CREATE TRIGGER massimo2passaggisegreti_trigger
  BEFORE INSERT ON dungeonsasdb.passaggio
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.massimo2passaggisegreti()
  ```

  Agisce prima di INSERT su Passaggio:

  ```
        -- provvede che ci siano al massimo 2 passaggi segreti IN ogni
  stanza
        -- provvede che ci siano al massimo 2 passaggi segreti VERSO una
           stanza
  ```

- ```
  CREATE TRIGGER updateinformazionigioco_trigger
  AFTER UPDATE OF visitedstanza_da
    ON dungeonsasdb.passaggio
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.updateinformazionigioco()
  ```

  Agisce dopo UPDATE di visitedStanza_DA su Passaggio:

  ```
        -- aggiorna l'attributo visited su InformazioniGioco (ovvero il
  numero di stanze uniche visitate dal personaggio nel gioco)
  ```

- ```
  CREATE TRIGGER cambiastanza_trigger
  AFTER UPDATE OF stanza_pg
    ON dungeonsasdb.personaggio
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.cambiastanza()
  ```

  Agisce dopo UPDATE di stanza_pg su Personaggio:

  ```
        -- cancella le tuple associate agli oggetti di tipo "pozione"
  dalla relazione Equipaggia
        -- aggiorna l'attributo visitedStanza_DA su Passaggio quando il
  personaggio si sposta
  ```

- ```
  CREATE TRIGGER gestioneinventario_trigger
  AFTER INSERT OR DELETE OR UPDATE ON dungeonsasdb.possiede
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.gestioneinventario()
  ```

  Agisce dopo INSERT o DELETE o UPDATE su Possiede

  Se INSERT o UPDATE:

  ```
        -- aggiorna l'attributo NR_OGG su Personaggio
        -- genera un eccezione se il numero degli oggetti posseduti supera
  il valore di ceil(COST/2)
        -- cancella dal mondo degli oggetti del personaggio(RandomOggetto)
  la tupla relativa all'oggetto aggiunto su Possiede(ovvero            quando il
  personaggio prende un oggetto dal mondo(RandomOggetto)         e lo mette
  nell'inventario(Possiede))
  ```

  Se DELETE:

-- aggiorna NR_OGG su Personaggio
-- cancella da Equipaggia gli oggetti che non sono ne pozione ne razione (in questo modo, dopo l'utilizzo, un oggetto consumabile può essere cancellato dall'inventario, ovvero Possiede, ma non viene cancellato da Equipaggia, perciò i bonus rimangono)

- **CREATE TRIGGER ultimaarma_trigger**
  AFTER DELETE ON dungeonsasdb.possiede
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.ultimaarma()

  Agisce dopo DELETE su Possiede, ovvero quando elimina un oggetto dal suo inventario:

  -- genera un'eccezione se l'oggetto di tipo arma che viene cancellato è l'ultimo oggetto di quel tipo nell'inventario (il personaggio non può eliminare tutte le armi che possiede altrimenti non potrebbe proseguire nell'avventura perchè per poter attaccare un nemico bisogna avere equipaggiato un'arma)

- **CREATE TRIGGER aggiornastanzanemici_trigger**
  AFTER INSERT OR DELETE ON dungeonsasdb.randomnemico
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.aggiornastanzanemici()

  Agisce dopo INSERT o DELETE su RandomNemico:

  -- Se INSERT: aumenta di 1 il valore di NR_NEM su Stanza
  -- Se DELETE: diminuisce di 1 il valore di NR_NEM su Stanza

- **CREATE TRIGGER balancenemici_trigger**
  AFTER INSERT ON dungeonsasdb.randomnemico
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.balancenemici()

  Agisce dopo INSERT su RandomNemico

  -- provvede che le stanze generate per il personaggio non siano troppo difficili, o addiritura impossibili, da affrontare.

  Esempio:

  - il personaggio inizia l'avventura con 10 punti in Attacco
  - il nemico inserito nel suo mondo ha 20 punti in Difesa
  - per poter attaccare con successo, il personaggio dovrebbe avere un valore A = (10 - 20) + tiro(1d20) > 12 ma ciò è impossibile perchè servirebbe un tiro del dado di valore > 22

  Al fine di evitare di generare stanze impossibili o troppo difficili ho introdotto questo trigger che diminuisce il valore DIF_N su RandomNemico se la disparità tra l'attacco del personaggio e la difesa del nemico è troppo alta. Il valore di difesa del nemico va diminuito secondo questo algoritmo:

  A = (attPG - difN) + 16 (ho scelto 16 perchè è valore abbastanza difficile da ottenere con il tiro di 1d20)

  dove attPG = attacco del personaggio, difN = difesa del nemico

IF (A < 12) THEN DIF_N = DIF_N - (12 - A)

Secondo l'esempio riportato sopra ho:

A = (10 - 20) + 16 = 6

Siccome ho (6 < 12) THEN DIF_N = 20 - (12 - 6) = 14;

Il nuovo valore DIF_N del nemico è 14, il che significa che il personaggio, per poterlo attaccare con successo, dovrebbe tirare 1d20 e ottenere un valore maggiore di A = 12 - (10 - 14) = 16 (quindi è abbastanza difficile da ottenere)

- **CREATE TRIGGER massimo2nemici_trigger**
  BEFORE INSERT ON dungeonsasdb.randomnemico
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.massimo2nemici()

  Agisce prima di INSERT su RandomNemico:

  -- provvede che ci siano al massimo 2 nemici per ogni stanza nel mondo del personaggio

- **CREATE TRIGGER mortenemico_trigger**
  AFTER UPDATE OF pf_n
  ON dungeonsasdb.randomnemico
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.mortenemico()

  Agisce dopo UPDATE di PF_N su RandomNemico

  -- Se il valore di PF_N di un nemico nel mondo del personaggio raggiunge il 0 (o valori negativi) allora il nemico va eliminato dal mondo. Inoltre dopo l'eliminazione viene aumentato di uno l'attributo defeated su InformazioniGioco e viene aumentato il valore dell'attributo danno_N_defeated su InformazioniGioco con un valore pari al danno del nemico sconfitto.

- **CREATE TRIGGER aggiornastanzaoggetti_trigger**
  AFTER INSERT OR DELETE ON dungeonsasdb.randomoggetto
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.aggiornastanzaoggetti()

  Agisce dopo INSERT o DELETE su RandomOggetto:

  -- Se INSERT: aumenta di 1 il valore di NR_OGG su Stanza
  -- Se DELETE: diminuisce di 1 il valore di NR_OGG su Stanza

- **CREATE TRIGGER massimo3oggetti_trigger**
  BEFORE INSERT ON dungeonsasdb.randomoggetto
  FOR EACH ROW EXECUTE PROCEDURE dungeonsasdb.massimo3oggetti()

  Agisce prima di INSERT su RandomOggetto:

  -- provvede che ci siano al massimo 3 oggetti in ogni stanza nel mondo del personaggio

Functions

- **CREATE FUNCTION *attacca* (character varying, character varying) RETURNS integer**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1
  -- Nome del nemico ALIAS FOR $2

  OUTPUT (INTEGER):
  -- 0 se nessuno è riuscito nell'attacco
  -- 1 se soltanto il personaggio è riuscito nell'attacco
  -- 2 se soltanto il nemico è riuscito nel'attacco
  -- 3 se entrambi sono riusciti negli attacchi

- **CREATE FUNCTION *calcolape* (character varying) RETURNS integer**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  OUTPUT (INTEGER):
  -- Valore dei punti d'esperienza raccolti durante il gioco

- **CREATE FUNCTION *cercanascosti* (character varying) RETURNS character varying**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  OUTPUT (VARCHAR(20)):

  -- Il nome dell'oggetto nascosto o del passaggio segreto trovato

- **CREATE FUNCTION *creapg* (integer, character varying, double precision, double precision, double precision, double precision) RETURNS void**

  INPUT:
  -- ID del Utente ALIAS FOR $1
  -- Nome del personaggio ALIAS FOR $2
  -- Valore di FOR ALIAS FOR $3
  -- Valore di INT ALIAS FOR $4
  -- Valore di AGI ALIAS FOR $5
  -- Valore di COST ALIAS FOR $6

  NO OUTPUT

  Crea un personaggio (oppure aggiorna se il personaggio esiste già) con i valori inseriti negli argomenti della funzione, richiama le funzioni generaNemici(nomepg), generaOggetti(nomepg) e generaPassaggi(nomepg) e infine inserisce nell'inventario del PG gli oggetti "Spada Base" e "Pane".

- **CREATE FUNCTION** ***datipg*** **(character varying) RETURNS** dungeonsasdb.datipersonaggio

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  OUTPUT (datiPersonaggio) - Gli object types sono elencati alla fine:
  -- Record di tipo datiPersonaggio contenente alcuni dati del PG

- **CREATE FUNCTION** ***diceroll*** **(integer, integer) RETURNS integer**

  INPUT:
  -- Numero di tiri di dado ALIAS FOR $1
  -- Valore del dado ALIAS FOR $2

  esempio: diceroll(3,6) tira 3 dadi da 6 e ne restituisce la somma

  OUTPUT (INTEGER):
  -- Valore complessivo dei tiri di dado

- **CREATE FUNCTION** ***finegame*** **(character varying) RETURNS void**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  NO OUTPUT

  Azzera il mondo e le statistiche di un personaggio.

  In particolare cancella tutte le tuple riferite al PG da Possiede, Equipaggia, RandomNemico, RandomOggetto, Passaggio. Inoltre imposta a DEFAULT tutte le statistiche del PG (tranne i punti esperienza) e imposta a DEFAULT la tupla relativa al PG nella tabella InformazioniGioco.

- **CREATE FUNCTION** ***generanemici*** **(character varying) RETURNS void**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  NO OUTPUT

  Inserisce nemici nel mondo del personaggio. La stanza è scelta casualmente.

- **CREATE FUNCTION** ***generaoggetti*** **(character varying) RETURNS void**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

NO OUTPUT

Inserisce oggetti (visibili o nascosti) nel mondo del personaggio.
La stanza è scelta casualmente.
Infine cambia la visibilità di tutti gli oggetti di tipo razione (ho scelto di farle diventare visibili al fine di facilitare il gioco).

- **CREATE FUNCTION** ***generapassaggi*** **(character varying) RETURNS void**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  NO OUTPUT

  Inserisce nel mondo del personaggio un passaggio visibile per ogni stanza esistente tranne l'ultima. In tal modo ogni stanza ha almeno un passaggio e non esistono stanze irraggiungibili. Alla fine richiama la funzione generaPassaggiSegreti($1).

- **CREATE FUNCTION** ***generapassaggisegreti*** **(character varying) RETURNS void**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  NO OUTPUT

  Inserisce nel mondo del personaggio fino a 15 passaggi segreti generati casualmente. I passaggi devono rispettare alcune condizioni per poter essere inserite.

- **CREATE FUNCTION** ***getinventario*** **(character varying) RETURNS SETOF record**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  OUTPUT (SET DI RECORD):
  -- Record di ogni oggetto che il personaggio possiede

- **CREATE FUNCTION** ***getstanzapg*** **(character varying) RETURNS integer**

  INPUT:
  -- Nome del personaggio ALIAS FOR $1

  OUTPUT (INTEGER):
  -- La stanza in cui si trova in quel momento

- **CREATE FUNCTION** ***prendioggetto*** **(character varying, character varying)**
  RETURNS void

  INPUT:
  -- Nome del personaggio ALIAS FOR $1
  -- Nome dell'oggetto ALIAS FOR $2

  NO OUTPUT

  Il personaggio $ mette nel suo inventario (Possiede) l'oggetto $2
  L'oggetto viene rimosso dal mondo RandomOggetto del personaggio.

- **CREATE FUNCTION** ***tredasei*** **() RETURNS SETOF dungeonsasdb.lancidadi**

  NO INPUT

  OUTPUT (SET di lanciDadi) - Gli object types sono elencati alla fine
  -- 5 record di lanciDadi. Ogni record contiene un tiro di 1d20

  Richiama la funzione diceroll(3,6) per 5 volte e restituisce una tabella con i risultati.

Object types

- **CREATE TYPE** datiPersonaggio **AS** (arma_eq **varchar**(20), armatura_eq **varchar**(20), "FOR" **integer**, **int integer**, agi **integer**, **cost float**, att **integer**, dif **integer**, per **integer**, pf **integer**, pe **integer**, capienza **integer**);

- **CREATE TYPE** lanciDadi **AS** (tiro **INTEGER**, valore **INTEGER**);

- **CREATE DOMAIN** DungeonsAsDB.tipi_di_visibilita **AS VARCHAR**(8) **CHECK** (**VALUE IN** ('visibile', 'nascosto'));

- **CREATE DOMAIN** tipi_oggetti **AS VARCHAR**(8) **CHECK** (**VALUE IN** ('arma', 'armatura', 'gioiello', 'pozione', 'razione'));