PRISM Game Reviews - Database

versione manuale: 3.1

Indice:

1 - Introduzione

1.1 - Mini Abstract

Il database di supporto per il sito di recensioni "PRISM Game Reviews" pone al centro dell'attenzione le recensioni riguardanti i videogiochi.

1.2 - Scelte di Rappresentazione

La tabella `RECENSIONE` è il cuore del database, ed è strettamente correlata alla tabella `GIOCO`.

Per una questione strumentale, per separare logicamente alcuni concetti, ed al fine di non appesantire troppo la tabella `RECENSIONE` si è optato per la creazione di quest'ultima tabella collegandola tramite una relazione `(1,1)-(1,1)` con la tabella `GIOCO`.

Tale scelta mette in evidenza che ogni recensione deve avere al più un gioco ad essa attinente, ed ogni gioco deve essere legato ad *una ed una sola* recensione.

> torna all'indice <

2 - Tabelle

Segue la documentazione per ogni tabella MySQL del Database.

2.1 - `RECENSIONE`

Contiene le informazioni sulle recensioni.

2.1.1 - Attributi

- ID:
    - Valore intero di tipo `AUTO_INCREMENT` da 1 a 99.999.
    - Nonostante l'`AUTO_INCREMENT` via PHP dev'essere inserito manualmente.
- Titolo:
    - Titolo della Recensione.
    - Stringa di massimo 255 caratteri.
- Contenuto:
    - Corpo della Recensione.
    - Testo di massimo 65.535 caratteri.
- TempoLettura:
    - Indicazione temporale sul tempo di lettura della recensione.
    - Calcolato lato server considerando una velocità di lettura media di 180 parole al minuto.

    - Valore intero di 3 cifre (approssimato per eccesso).
    - Non va inserito manualmente.
- Keywords:
    - Stringa di parole chiave *separate da virgole* per la generazione e l'indicizzazione dell'HTML.
    - Testo di massimo 1.000 caratteri.
    - Può contenere fino a 10 keywords in ordine decrescente d'importanza, alcune delle quali sono standard per ogni pagina del sito, le altre peculiari di ogni recensione e videogioco ad essa associato sono parametriche e calcolate via PHP (vedi note).
    - Non va inserito manualmente.
- DescrizioneHTML:
    - Breve descrizione/sinossi della recensione, per la generazione e l'indicizzazione dell'HTML.
    - Stringa di massimo 150 caratteri.
- Autore:
    - Username dell'amministratore autore della recensione.
    - Foreign key con `UTENTE(Username)`.
    - Non si può eliminare un amministratore se ha redatto almeno una recensione.
    - Se il nome di un amministratore viene aggiornato si aggiorna anche in `RECENSIONE`.
    - Non va inserito manualmente.
- AutoreModifica:
    - Username dell'amministratore autore dell'ultima modifica alla recensione.
    - Ha valore `NULL` di default fino alla prima modifica.
    - Foreign key con `UTENTE(Username)`.
    - Non si può eliminare un amministratore se ha modificato almeno una recensione.
    - Se il nome di un amministratore viene aggiornato si aggiorna anche in `RECENSIONE`.
    - Non va inserito manualmente.
- DataPubblicazione:
    - Data della prima pubblicazione di una recensione.
    - Di tipo `DATE` nel formato americano `AAAA-MM-GG`.
    - Non va inserito manualmente.
- DataModifica:
    - Data dell'ultima modifica apportata alla recensione.
    - Ha valore `NULL` di default fino alla prima modifica.
    - Di tipo `DATE` nel formato americano `AAAA-MM-GG`.
    - Non va inserito manualmente.

2.1.2 - Chiavi e Vincoli

- Primary key: (ID)
- Può avere *zero, uno o più* `COMMENTI`.
- Si riferisce ad *uno ed un solo* `GIOCO`.

2.1.3 - Note

- Una recensione *deve* essere redatta da un amministratore (via PHP).
- Una recensione *può* essere modificata da un amministratore (via PHP).
- La prima keyword sarà il titolo del gioco, seguiranno la seconda la parola "recensione", lo sviluppatore del gioco, l'anno di uscita del gioco, la parola "videogioco", la parola "review", la parola "game" e per ultima la parola "Prism". (Massima lunghezza teorica: 255 titoloGioco + 255 sviluppatore + 4 annoUscita + 7 virgole + 35 keywords aziendali = 556. Si noti inoltre che prima dell'inserimento verranno rimossi eventuali tag html validi presenti nei campi utilizzati, come titoloGioco, e quindi la lunghezza reale è minore).

> torna all'indice <

2.2 - `GIOCO`

Contiene le informazioni sui videogiochi, in questo contesto solo strumentali, pertanto la loro esistenza è legata con quella della loro `RECENSIONE`.

2.2.1 - Attributi

- IDGioco:
    - Foreign key con `RECENSIONE(ID)`.
    - Se cancellato una recensione si cancella anche il `GIOCO` cui si riferisce.
    - Calcolato e inserito via PHP.
- Titolo:
    - Titolo del gioco.
    - Stringa di massimo 255 caratteri.
- TitoloOrdinamento:
    - Titolo del gioco senza tag HTML.
    - Stringa di massimo 255 caratteri.
    - Necessario per l'ordinamento dei risultati via PHP.

    - Non va inserito manualmente.
- CoverExt:
    - Contiene l'estensione del file della copertina del gioco, necessario per il MIME nell'HTML.
    - Non va inserito manualmente.
- CoverDescr:
    - La descrizione per la "figcaption" della copertina.
    - Stringa di massimo 255 caratteri.
- Sviluppatore:
    - Nome dello sviluppatore o della casa di produzione autrice del gioco.
    - Stringa di massimo 255 caratteri.
- AnnoUscita:
    - Anno di uscita della versione internazionale del gioco.
    - Valore intero di 4 cifre.
- SitoUfficiale:
    - URL del sito web di riferimento (default "ND").
    - Stringa di massimo 128 caratteri.
- Pegi:
    - Indicatore dell'età target del gioco.
    - Valore di tipo `ENUM`: `3+`, `7+`, `12+`, `16+` e `18+`.

2.2.2 - Chiavi e Vincoli

- Primary key: (IDGioco)
- Unique: (Titolo, AnnoUscita)
- Appartiene (vedi `APPARTENENZA`) ad *uno o più* `GENERI`
- È eseguibile (vedi `ESECUZIONE`) su *una o più* `PIATTAFORME`

2.2.3 - Note

- L'attributo IDGioco deve corrispondere all'attributo ID di `RECENSIONE`, pertanto va prima inserita la recensione poi recuperato l'ID e poi inserito il gioco.
- L'attributo TitoloOrdinamento è ridondante (le informazioni sono le stesse dell'attributo Titolo), è stato inserito per rendere più agevole il recupero dell'informazione ordinata alfabeticamente.

> torna all'indice <

2.3 - `GENERE`

Contiene i generi a cui può appartenere un `GIOCO`.

2.3.1 - Attributi

- IDGenere:
    - Valore intero di tipo `AUTO_INCREMENT` da 1 a 99.
    - Non va inserito manualmente.
- Nome:
    - Nome del genere.
    - Stringa di massimo 128 caratteri.
    - Esempi: Azione, Manageriale, Sparatutto, Sportivo, ecc.

2.4.2 - Chiavi e Vincoli

- Primary key: (IDGenere)
- Unique: (Nome)

2.4.3 - Note

- Un gioco deve avere *uno o più* generi. (vedi `APPARTENENZA`).
- Un genere può appartenere a *zero, uno o più* giochi (vedi `APPARTENENZA`).
- La `INSERT` in `GENERE` è possibile solo manualmente con accesso diretto al DB e non tramite UI o form.

> torna all'indice <

2.4 - `APPARTENENZA`

D'implementazione, nasce dalla relazione `N-N` fra `GIOCO` e `GENERE`.

2.4.1 - Attributi

- IDGioco:
  - Foreign key con `GIOCO(IDGioco)`.
  - Se cancellato un gioco si cancellano anche i suoi record in `APPARTENENZA`.
- IDGenere:
  - Foreign key con `GENERE(IDGenere)`.
  - Non si può eliminare un genere se appartiene ad almeno un gioco.

2.4.2 - Chiavi e Vincoli

Primary key: (IDGioco, IDGenere)

2.4.3 - Note

- Ogni `GIOCO` deve avere *uno o più* generi.
- Un `GENERE` può appartenere a *zero, uno o più* giochi.
- La `INSERT` in `APPARTENENZA` è automatica.

> torna all'indice <

2.5 - `PIATTAFORMA`

Contiene le piattaforme per cui è disponibile un `GIOCO`.

2.5.1 - Attributi

- IDPiattaforma:
  - Valore intero di tipo `AUTO_INCREMENT` da 1 a 99.
  - Non va inserito manualmente.
- Nome:
  - Stringa di massimo 64 caratteri.
  - Esempi: XBox, PC, PlayStation
- Versione:
  - Stringa di massimo 64 caratteri.
  - Esempi: 360, One, XP, 3, 4, ecc.
- AnnoRilascio:
  - Anno di rilascio di una particolare versione di piattaforma.
  - Valore intero di 4 cifre.

2.5.2 - Chiavi e Vincoli

- Primary key: (IDPiattaforma)
- Unique: (Nome, Versione)

2.5.3 - Note

- Un `GIOCO` deve avere *una o più* piattaforme. (vedi `ESECUZIONE`).
- Una piattaforma può eseguire *zero, uno o più* `GIOCHI` (vedi `ESECUZIONE`).
- La `INSERT` in `PIATTAFORMA` è possibile solo manualmente con accesso diretto al DB e non tramite UI o form.

> torna all'indice <

2.6 - `ESECUZIONE`

D'implementazione, nasce dalla relazione `N-N` fra `GIOCO` e `PIATTAFORMA`.

2.6.1 - Attributi

- IDGioco:
  - Foreign key con `GIOCO(IDGioco)`.
  - Se cancellato un gioco si cancellano anche i suoi record in `ESECUZIONE`.
- IDPiattaforma:
  - Foreign key con `PIATTAFORMA(IDPiattaforma)`.
  - Non si può eliminare una versione di una piattaforma se esegue almeno un gioco.

2.6.2 - Chiavi e Vincoli

Primary key: (IDGioco, IDPiattaforma)

2.6.3 - Note

- Ogni `GIOCO` deve essere eseguito su *una o più* versioni di *una o più* `PIATTAFORME`.
- Una versione di una `PIATTAFORMA` può eseguire *zero, uno o più* `GIOCHI`.
- La `INSERT` in `ESECUZIONE` è automatica.

> torna all'indice <

2.7 - `UTENTE`

Contiene i dati degli utenti registrati (sia *Amministratori* sia *Standard*).

2.7.1 - Attributi

- Email:
  - Email dell'utente.
  - Stringa di massimo 128 caratteri.
  - `UNIQUE` non ci possono essere email duplicate.
- Username:
  - Primary key, identifica univocamente l'utente ove necessario.
  - Stringa di massimo 32 caratteri.
- DataIscrizione:
  - Data di iscrizione dell'utente.
  - Di tipo `DATE` nel formato americano `AAAA-MM-GG`.
  - Non va inserito manualmente.
- HashPassword:
  - Stringa di massimo 255 caratteri contenente l'hash della password.
- Administrator:
  - Valore booleano, `true` se è un amministratore, `false` se è uno standard_user.
  - Default: `false`.
- Eliminato:
  - Valore Booleano, `true` se è un amministratore eliminato, `false` in tutti gli altri casi.
  - Se `Eliminato` è `true`, allora `Username` sarà della forma `Username_Eliminato`.
  - Default: `false`.

2.7.2 - Chiavi e Vincoli

- Primary key: (Username)
- Unique: (Email)

2.7.3 - Note

- HashPassword: la password arriva in chiaro, ne viene calcolato l'hash lato server, e questo è inserito nel DB (via PHP).
- La `INSERT` in `UTENTE` aggiunge solamente nuovi utenti standard.
- La `INSERT` in `UTENTE` con l'attributo `Administrator` impostato a `true` è possibile solo manualmente con accesso diretto al DB e non tramite UI o form anche se *sconsigliata*, si usi invece l'apposita procedura (PROMOTE_USER).
- La `DELETE` su `UTENTE` è *sconsigliata*, si usi invece l'apposita procedura (DELETE_USER);
  - L'eliminazione di un `UTENTE` con l'attributo `Administrator` impostato a `false` elimina il record e tutti i suoi eventuali `COMMENTI` (lo username torna disponibile);
  - L'eliminazione di un `UTENTE` con l'attributo `Administrator` impostato a `true` comporta l'eliminazione di tutti i suoi eventuali `COMMENTI` ed imposta l'attributo `Eliminato` a `true` (lo username *non* torna disponibile).
- `Eliminato` è `true` soltanto se tale username apparteneva ad un amministratore ora eliminato, lo username è pertanto del tipo Username_Eliminato.

> torna all'indice <

2.8 - `COMMENTO`

D'implementazione, nasce dalla relazione `N-N` fra `RECENSIONE` e `UTENTE`, contiene i commenti degli utenti alle recensioni.

2.8.1 - Attributi

- IDRecensione:
  - Foreign key con `RECENSIONE(ID)`.
  - Se cancellata una recensione si cancellano anche i suoi record in `COMMENTO`.
- Utente:
  - Foreign key con `UTENTE(Username)`.
  - Se lo username di un utente viene eliminato o aggiornato tale azione si propaga anche a `COMMENTO`.
- Contenuto:
  - Testo di massimo 1.000 caratteri contenente il commento dell'utente.
- DataCommento:
  - Data di pubblicazione o ultima modifica del commento.

- Di tipo `DATE` nel formato americano `AAAA-MM-GG`.

2.8.2 - Chiavi e Vincoli

Primary key: (IDRecensione, Utente)

2.8.3 - Note

- La `INSERT` in `COMMENTO` necessita solo dell'attributo `Contenuto`, tutti gli altri sono o inviati dal client (`Utente`) o calcolati lato server (le date) o automatici di MySQL.
- Un utente può modificare *solo* i propri commenti.
- Un utente può commentare *zero, una o più* recensioni.
- Un utente può commentare *al più una volta* una data recensione, dopodichè può solo modificare il suo commento.
- Una recensione può avere *zero, uno o più* commenti.
- Se un utente (sia standard sia amministratore) si elimina tutti i suoi commenti vengono eliminati (la recensione non viene toccata) (via procedura).
- Se una recensione viene eliminata tutti i suoi commenti vengono eliminati, nulla accade agli utenti autori dei commenti.

> torna all'indice <

3 - Trigger

Segue la documentazione per i trigger del Database.

3.1 - DeleteRecensione

Status: non attivo

Se un `GIOCO` viene eliminato allora elimina anche la `RECENSIONE` ad esso associata.

3.2 - NotDeleteGioco

Status: attivo

Impedisce la cancellazione di un `GIOCO` se vi è collegata una `RECENSIONE`.

3.3 - PreventIllegalInsertIntoRecensione

Status: attivo

Impedisce inserimenti illegittimi in `RECENSIONE`, ovvero se:

- l'attributo Autore non corrisponde allo Username di un Amministratore,
- si tenta l'inserimento anche degli attributi AutoreModifica e/o DataModifica.

3.4 - PreventIllegalUpdateOnRecensione

Status: attivo

Impedisce modifiche illegittime a `RECENSIONE`, ovvero se:

- gli attributi AutoreModifica e/o DataModifica sono `NULL`,
- l'attributo AutoreModifica non è corrisponde allo Username di un Amministratore,
- si tenta la modifica degli attributi Autore e/o DataPubblicazione.

Nota: il trigger impedisce la *prima* modifica ad una `RECENSIONE` se non vengono esplicitamente forniti gli attributi AutoreModifica e DataModifica; *successive* modifiche (`UPDATE`) a tale `RECENSIONE` risultano legittime anche se non vengono esplicitamente forniti tali attributi.

> torna all'indice <

4 - Procedure

Segue la documentazione per le procedure del Database.

4.1 - DELETE_USER

Status: attiva

Sintassi: `CALL DELETE_USER("Username");`

Scopo: eliminare l'`UTENTE` identificato da "Username":

- se Amministratore elimina tutti i suoi eventuali commenti, e setta il campo Eliminato a true,

- altrimenti elimina il record da `UTENTE` (e per cascade si eliminano i suoi eventuali `COMMENTI`).

Nota: è bene eseguire eventuali eliminazioni sempre e solo mediante questa procedure; l'uso di `DELETE` è sconsigliato per garantire l'integrità del database con quanto dichiarato in questa documentazione.

4.2 - PROMOTE_USER

Status: attiva

Sintassi: `CALL PROMOTE_USER("Username");`

Scopo: promuovere l'`UTENTE` identificato da "Username" ad Amministratore.

> torna all'indice <

5 - Test

Serie di test per verificare la solidità e la correttezza del DB, la popolazione è quella disponibile nel file `database.sql`.

1. 01-
   - `DELETE FROM GENERE WHERE Nome='Azione';`
   - `DELETE FROM GENERE WHERE IDGenere=2;`
   - test: la cancellazione di un genere cui appartenga un gioco nel DB dev'essere impedita.
   - status: `OK`
2. 02-
   - `DELETE FROM GENERE WHERE Nome='Sportivo';`
   - `DELETE FROM GENERE WHERE IDGenere=3;`
   - test: la cancellazione di un genere cui nessun gioco appartiene deve avvenire.
   - status: `OK`
3. 03-
   - `DELETE FROM COMMENTO WHERE Utente='user' AND IDRecensione=1;`
   - test: si deve poter cancellare un commento di un utente (sia standard sia amministratore, basta questo test qui) ad una recensione, senza intaccare ne l'utente ne la recensione.
   - status: `OK`
4. 04-
   - `UPDATE UTENTE SET Username='user4' WHERE Username='user2';`
   - test: l'update dello username di un utente deve essere possibile e propagarsi nei suoi eventuali commenti.
   - status: `OK`
5. 05-
   - `UPDATE UTENTE SET Username='admin5' WHERE Username='admin';`
   - test: l'update dello username di un amministratore deve essere possibile e propagarsi nei suoi eventuali commenti e nelle recensioni da lui scritte e/o modificate.
   - status: `OK`
6. 06-
   - `DELETE FROM UTENTE WHERE Username='admin2';`
   - test: eliminando un amministratore vanno eliminati tutti i suoi eventuali commenti, va settato il flag booleano "Eliminato" a `true` nella tabella UTENTI, non vanno toccate eventuali recensioni da lui scritte o modificate, lo username non torna disponibile; se l'Amministratore è l'Autore (o l'AutoreModifica) di almeno una Recensione i cascade ne impediscono l'eliminazione (si consiglia di usare la procedura apposita, vedi Test 08).
   - status: `FAIL`
7. 07-
   - `DELETE FROM UTENTE WHERE Username='admin4';`
   - test: eliminando un amministratore vanno eliminati tutti i suoi eventuali commenti, va settato il flag booleano "Eliminato" a `true` nella tabella UTENTI, non vanno toccate eventuali recensioni da lui scritte o modificate, lo username non torna disponibile; se l'Amministratore *non* è l'Autore (o l'AutoreModifica) di almeno una Recensione vengono eliminati i suoi eventuali commenti ma viene eliminata la sua entry da UTENTI ed il suo Username torna disponibile (si consiglia di usare la procedura apposita, vedi Test 08).
   - status: `FAIL`
8. 08-
   - `CALL DELETE_USER('admin5');`
   - test: eliminando un amministratore vanno eliminati tutti i suoi eventuali commenti, va settato il flag booleano "Eliminato" a `true` nella tabella UTENTI, non vanno toccate eventuali recensioni da lui scritte o modificate, lo username non torna disponibile (via procedura apposita).
   - status: `OK`
9. 09-

- `DELETE FROM UTENTE WHERE Username='user2';`
- `CALL DELETE_USER('user2');`
- test: eliminando un utente standard vanno eliminati tutti i suoi eventuali commenti, va eliminata la sua entry nella tabella UTENTI, lo username torna disponibile (si consiglia di usare la procedura apposita).
- status: `OK`

10. 10-

    - `DELETE FROM GIOCO WHERE IDGioco=1;`
    - test: la cancellazione di un gioco deve essere impedita se ad esso è associata la sua recensione (via trigger).
    - status: `OK`

11. 11-

    - `DELETE FROM RECENSIONE WHERE ID=2;`
    - test: cancellando una recensione vanno eliminati tutti i suoi commenti (senza inficiare gli utenti standard o gli amministratori), va eliminato il gioco a cui si riferisce e di conseguenza le appartenenze del gioco ad un genere e le esecuzioni su una qualche piattaforma di tale gioco.
    - status: `OK`

12. 12-

    - `INSERT INTO RECENSIONE (Titolo, Contenuto, TempoLettura, Keywords, DescrizioneHTML, Autore, DataPubblicazione)`
      `VALUES ('DummyRec', 'DummyRecCont', 4, 'dummy keyw', 'DummyHTMLDesc', 'admin2', '2000-01-01');`
    - test: è permesso l'inserimento di una nuova recensione solo se Autore è lo username di un Amministratore e se *non* vengono specificati AutoreModifica e/o DataModifica diversi da `NULL` (via trigger).
    - status: `OK`

13. 13-

    - `INSERT INTO RECENSIONE (Titolo, Contenuto, TempoLettura, Keywords, DescrizioneHTML, Autore, AutoreModifica, DataPubblicazione, DataModifica)`
      `VALUES ('DummyRec', 'DummyRecCont', 4, 'dummy keyw', 'DummyHTMLDesc', 'admin3', 'admin2', '2001-01-01', '2001-02-01');`
    - test: è impedito l'inserimento di una nuova recensione se Autore è lo username di un Amministratore ma vengono specificati AutoreModifica e/o DataModifica diversi da `NULL` (via trigger).
    - status: `OK`

14. 14-

    - `INSERT INTO RECENSIONE (Titolo, Contenuto, TempoLettura, Keywords, DescrizioneHTML, Autore, DataPubblicazione)`
      `VALUES ('DummyRec', 'DummyRecCont', 4, 'dummy keyw', 'DummyHTMLDesc', 'user', '2002-01-01');`
    - test: è impedito l'inserimento di una nuova recensione se Autore *non* è lo username di un Amministratore (via trigger).
    - status: `OK`

15. 15-

    - `UPDATE RECENSIONE SET Titolo='DummyRec2', TempoLettura=3, AutoreModifica='admin3', DataModifica='2000-02-01' WHERE ID=1;`
    - test: è permessa la modifica di una recensione solo se AutoreModifica è lo username di un Amministratore, se è specificata DataModifica diversa da `NULL` e se *non* vengono modificati Autore e/o DataPubblicazione (via trigger).
    - status: `OK`

16. 16-

    - `UPDATE RECENSIONE SET Titolo='DummyRec2', AutoreModifica=NULL WHERE ID=1;`
    - test: è impedita la modifica di una recensione se AutoreModifica e/o DataModifica sono `NULL` (via trigger).
    - status: `OK`

17. 17-

    - `UPDATE RECENSIONE SET Titolo='DummyRec2', AutoreModifica='user', DataModifica='2001-02-01' WHERE ID=1;`
    - test: è impedita la modifica di una recensione se AutoreModifica *non* è lo username di un Amministratore (via trigger).
    - status: `OK`

18. 18-

    - `UPDATE RECENSIONE SET Titolo='DummyRec2', Autore='admin2', AutoreModifica='admin3', DataModifica='2002-02-01' WHERE ID=1;`
    - test: è impedita la modifica di una recensione se vengono modificati Autore e/o DataPubblicazione (via trigger).
    - status: `OK`

19. 19-

    - `UPDATE RECENSIONE SET Titolo='DummyRec2' WHERE ID=1;`
    - test: si vorrebbe impedire una modifica se non vengono esplicitamente indicati gli attributi AutoreModifica e DataModifica, ma ciò è possibile solo se suddetti attributi sono `NULL` nel DB prima dell'update (vedi nota del trigger dedicato).
    - status: `FAIL`

20. 20-

    - `CALL PROMOTE_USER('newAdmin');`
    - test: si vuole impostare a `true` l'attributo Administrator dell'utente newAdmin.
    - status: `OK`

Status complessivo: 17`OK` - 0`UNCHECKED` - 3`FAILED`

> torna all'indice <