

Progetto di Basi di Dati

Inventario Virtuale

13/02/2024

Mattia Michel Bonaccio

Università degli studi di Salerno

0512108091

Panoramica

Si vuole progettare una base di dati per la gestione di un inventario virtuale per la una piattaforma di videogiochi.

Un utente può accedere alla piattaforma attraverso l'uso di un username univoco ed una password. Una volta effettuato l'accesso sarà disponibile la gestione del proprio inventario e dei relativi oggetti contenuti in esso.

Gli oggetti sono legati ai videogiochi disponibili sulla piattaforma e possono essere di vari tipi. Ogni gioco presenta una lista di oggetti associati. Gli oggetti, inoltre, possono variare nella loro rarità e in casi particolari, nel livello di usura.


Gli oggetti possono essere messi in vendita ad un prezzo determinato dall'utente, per poi essere acquistati da altri.

Specifiche della realtà d'interesse

Il progetto punta a rappresentare una piattaforma di gestione di oggetti ed equipaggiamenti virtuali da utilizzare in relativi videogiochi o semplicemente da collezionare. L'obiettivo è di fornire agli utenti un sistema in cui è possibile scambiare, acquistare e vendere i numerosi "Item" in modo organico interagendo con la community presente.

L'idea di un inventario virtuale è sempre più popolare nel mondo videoludico, offrendo un sistema dove i giocatori possono collezionare svariati tipi di oggetti. Partendo dalle classiche carte collezionabili dei loro giochi preferiti che, una volta accumulate in numero sufficiente, è possibile scambiare per un pacchetto di equipaggiamento. Questi pacchetti sono di vitale importanza per la longevità del sistema, poiché da essi, una volta aperti, l'utente riceverà nel proprio inventario, un nuovo equipaggiamento utilizzabile in gioco.

Infatti, uno degli aspetti fondamentali per cui questo tipo di soluzione è di grande successo è proprio la possibilità di espressione individuale attraverso il proprio avatar. Ogni utente può personalizzare il proprio avatar all'interno di un gioco utilizzando gli equipaggiamenti provenienti dal proprio inventario in modo da



esprimere il suo stile, creando un ambiente sociale dinamico, in cui i videogiocatori interagiscono anche oltre la semplice partita.

Saranno disponibili diversi mezzi con i quali un utente può ricevere nuovi oggetti. Per quanto riguarda le carte collezionabili, queste verranno ricevute dai giocatori ogni manciata di ore di gioco, attraverso l'accumulo di punti; Una volta completata una collezione di carte, un utente può decidere di convertirla in un pacchetto di equipaggiamento. Una volta aperto, il giocatore riceverà un pezzo di equipaggiamento del gioco a cui appartiene il pacchetto.

Oltre alla conversione di carte in pacchetti, gli utenti possono ricevere nuovi items acquistando dal mercato integrato. Sarà infatti possibile effettuare una ricerca negli inventari della community inserendo i parametri degli oggetti a cui si è interessati, come la collezione da cui provengono, la loro rarità, il gioco, e il livello di usura.

Una volta trovato un oggetto a cui si è interessati, un utente può effettuare l'acquisto. L'oggetto verrà trasferito dall'inventario del venditore a quello dell'acquirente mantenendo le caratteristiche originali.

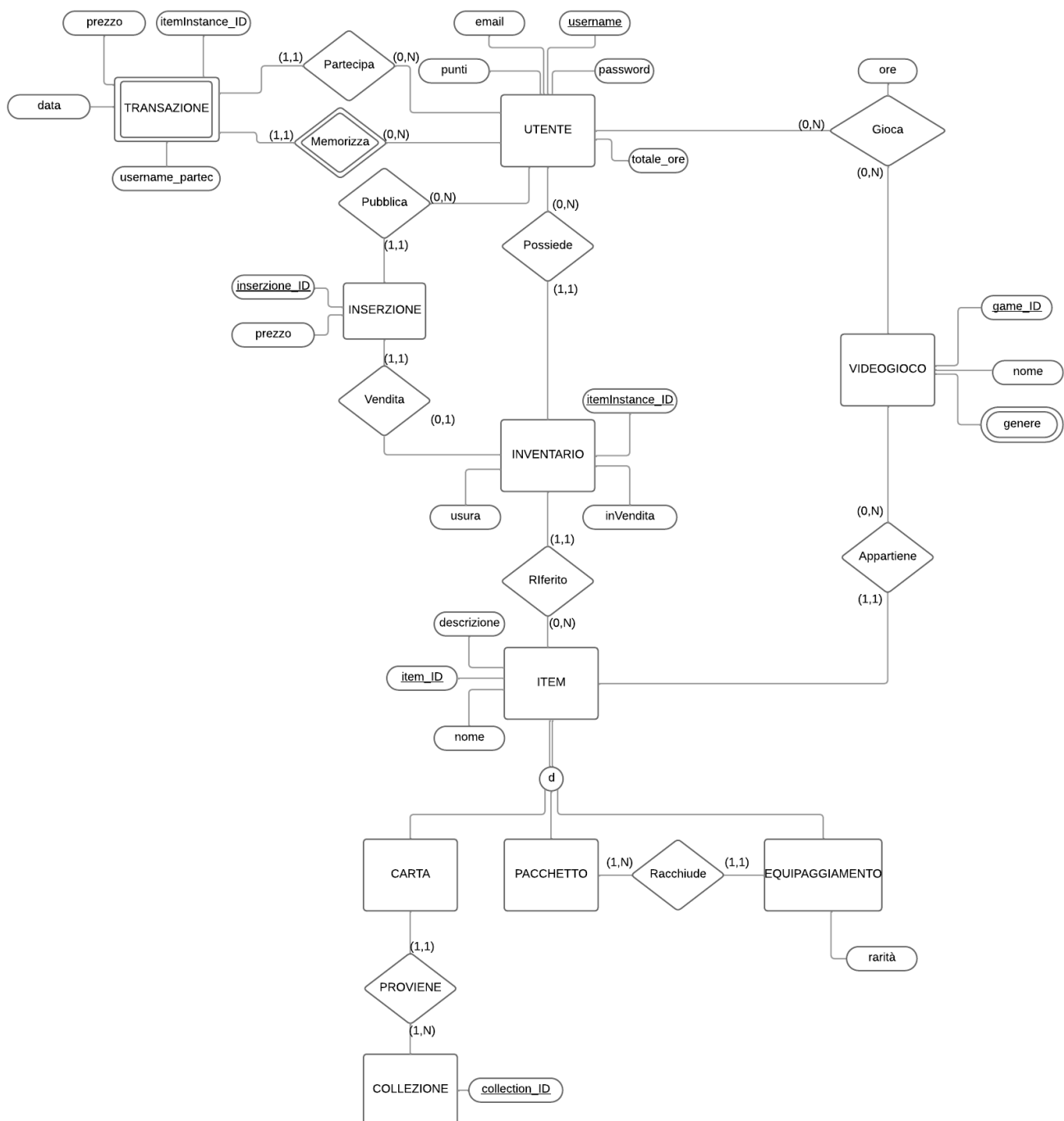
Gli oggetti sono gestiti dalla piattaforma nel seguente modo: viene inizialmente memorizzata una "pool" di Item per i videogiochi che li implementano. Questi item possono essere carte collezionabili, pacchetti apribili, e pezzi di equipaggiamento. Da questo insieme di item, un utente riceve una "istanza" di item, con un proprio codice identificativo, ed un valore che indica il livello di usura (oggetti con livelli di usura particolari potrebbero risultare più "desiderabili" dai giocatori)

Glossario dei termini

Termine	Significato
UserAccount	Il profilo con il quale ogni utente può accedere al servizio.
Inventario	L'area riferita ad un account dove è possibile conservare i propri oggetti virtuali. Qui sono memorizzate le istanze di item veri e propri, che possono essere usati in gioco o venduti sul mercato.
Gioco	Videogioco disponibile agli utenti, che possono effettuare delle sessioni di gioco di una certa durata.
Item	Oggetto generico appartenente ad un videogioco. Può essere una carta collezionabile, un pacchetto o un equipaggiamento. è sempre riferito ad un gioco. Si usa per indicare il "prototipo" di un item, che non è ancora posseduto da un giocatore.
Carta	Carta collezionabile ottenibile attraverso lo scambio di punti accumulabili con ore di gioco. Caratterizzato anche da un indicatore di usura.
Equipaggiamento	Oggetto ottenibile attraverso l'apertura di pacchetti. Caratterizzato da un livello di rarità e un indicatore di usura.
Pacchetto	Oggetto ottenibile convertendo un numero specifico di carte. Alla sua apertura l'utente riceverà un equipaggiamento casuale dalla "pool" contenuta in esso.
Pool	Si usa per indicare le possibili ricompense ottenibili da un pacchetto. Nello schema ER è rappresentato con la relazione "Racchiude"
Punti	Punti che un utente accumula effettuando sessioni di gioco. Ogni ora di gioco equivale a 10 punti.
Inserzione	Annuncio con il quale un giocatore può mettere in vendita un oggetto da lui posseduto ad un determinato prezzo.

Progettazione concettuale della base di dati

Schema EER



Dizionario delle entità

Legenda: sotto-entità, attributo multivalore, attributo ridondante, entità debole, chiave candidata

Entità	Descrizione	Attributi	Identificatore
Utente	Profilo di un utilizzatore della piattaforma	<ul style="list-style-type: none"> - username - email - password - punti - totale_ore 	username
Inventario	Insieme di oggetti posseduti da un utenti	<ul style="list-style-type: none"> - ItemInstance_ID - usura - inVendita 	ItemInstance_ID
Item	Prototipo di oggetto collezionabile	<ul style="list-style-type: none"> - item_ID - nome - descrizione 	item_ID
Videogioco	Software a disposizione dell'utente	<ul style="list-style-type: none"> - game_ID - nome - genere 	game_ID
Inserzione	Annuncio di vendita di un oggetto da parte di un utente	<ul style="list-style-type: none"> - inserzione_ID - prezzo 	inserzione_ID
Transazione	Vendite passate di oggetti memorizzate per ogni utente	<ul style="list-style-type: none"> - prezzo - data_ora - itemInstance_ID - username_partec 	
Carta	Tipo di Item, viene aggiunta all'inventario di un utente al raggiungimento di un certo numero d'ore. Appartiene ad una collezione.		
Pacchetto	Tipo di Item, contiene		

	un Equipaggiamento casuale che viene rivelato solo all'apertura da parte di un utente. Ottenibile scambiando una collezione completa di carte.		
Equipaggiamento	Tipo di Item, è un oggetto utilizzabile nel gioco da cui proviene.	- rarità	
Collezione	Un insieme specifico di carte. Una collezione è formata da 5 carte collezionabili. Una volta completata la collezione, un utente può scambiarla attraverso la piattaforma per ricevere un pacchetto.	- collection_ID	collection_ID

Dizionario delle relazioni

Relazione	Descrizione	Entità coinvolte	Attributi
Gioca	Un utente gioca a dei videogiochi che possiede. "ore" indica il numero di ore accumulate da un utente per un gioco.	Utente(0,N) Videogioco(0,N)	ore
Appartiene	Un Item appartiene ad uno specifico gioco. Nel caso degli equipaggiamenti, questi possono essere utilizzati nel videogioco interessato.	Item(1,1) Videogioco(0,N)	

Possiede	Un'istanza di un item è posseduta da un utente.	Utente(0,N) Inventario(1,1)	
Riferito	Un'istanza di un item è riferita ad un prototipo di item.	Inventario(1,1) Item(0,N)	
Pubblica	Un utente può pubblicare un'inserzione per vendere un Item sul mercato.	Utente(0,N) Inserzione(1,1)	
Vendita	L'inserzione è riferita ad un'istanza di item.	Inserzione(1,1) Inventario(0,1)	
Racchiude	Un pacchetto può contenere un equipaggiamento casuale.	Pacchetto(1,N) Equipaggiamento(1,1)	
Proviene	Un insieme di carte forma una collezione	Carta(1,1) Collezione(1,N)	
Memorizza	Le vendite passate sono memorizzate per ogni utente	Utente(0,N) Transazione(1,1)	
Partecipa	Indica la partecipazione dell'acquirente nella transazione	Utente(0,N) Transazione(1,1)	

Vincoli non esprimibili nello schema

- L'usura di un equipaggiamento è un valore in virgola mobile maggiore di 0 e minore di 1.
- La rarità è un campo che può assumere i valori "Comune", "Raro", "Epico" e "Leggendario"
- Il prezzo di un'inserzione deve essere compreso tra 0.04€ e 100€ .

Definizione delle procedure per la gestione della base di dati

Tavola dei volumi

Concetto	Tipo	Carico Applicativo
Utente	E	50
Inventario	E	100
Item	E	75
Videogioco	E	10
Inserzione	E	30
Carta	E	50
Pacchetto	E	5
Equipaggiamento	E	100
Collezione	E	10
Gioca	R	250
Appartiene	R	75
Possiede	R	100
Pubblica	R	75
Vendita	R	150
Racchiude	R	100
Proviene	R	50

Tavola delle operazioni

	Operazione	Tipo	Frequenza
1	Creazione utente	I	10/mm
2	Utente gioca n ore ad un videogioco	I	40/gg
3	Aggiungi un gioco alla piattaforma	I	1/mm
4	Crea un'inserzione	I	15/gg
5	Utente apre un pacchetto ed ottiene un equipaggiamento	I	10/gg
6	Stampa gli oggetti posseduti da ogni utente	B	1/gg

7	Utente scambia 5 carte collezionabili per un pacchetto	I	15/gg
8	Utente riscatta una carta consumando 20 punti	I	5/gg
9	Stampare tutti gli oggetti di un utente con rarità "leggendaria"	B	40/gg
10	Cerca gli utenti che hanno giocato ad almeno 5 videogiochi	B	5/aa
11	Inserire un nuovo pacchetto ed aggiungere un oggetto al suo contenuto	I	4/mm
12	Inserire un oggetto in un pacchetto già esistente	I	10/mm
13	Cercare tutte le inserzioni per un determinato Item	I	15/gg
14	Stampare la lista e il numero di videogiochi giocati di tutti gli utenti in ordine crescente di videogiochi giocati	B	10/mm
15	Stampare la lista e il numero di ore totali di tutti gli utenti in ordine crescente di ore di gioco totali	B	5/mm
16	Cercare l'utente con il maggior numero di Item nel proprio inventario	B	1/gg
17	Utente effettua un acquisto di un Item	I	2/gg
18	Seleziona gli utenti che hanno almeno 5 oggetti ed una trasazione.	B	5/gg
19	Seleziona gli utenti che giocano a "Dota 2" e possiedono oggetti di rarità "Leggendaria"	B	1/mm

Progettazione logica

Analisi delle ridondanze

Nello schema EER è presente un attributo ridondante, "totale_ore" dell'entità "Utente". Questo attributo è derivabile dalla relazione "Gioca", sommando il numero di ore relative ad uno specifico utente. L'attributo "totale_ore" verrebbe memorizzato con un valore di tipo float, occupando 4 byte per singolo utente quindi, considerando un carico applicativo di 50 utenti, il costo totale in memoria sarà di 200 byte.

Si prosegue quindi con l'analisi del numero di accessi delle operazioni che riguardano questo attributo, per valutare se mantenere o meno la ridondanza.

Tavole degli accessi

Analisi delle operazioni con ridondanza

Operazione 2				Operazione 15			
Tabella	Tipo	Accessi	Tipo ac.	Tabella	Tipo	Accessi	Tipo ac.
Gioca	R	1	L	Utente	E	50	L
Gioca	R	1	S				
Utente	E	1	L				
Utente	E	1	S				
Totale: $[2 + (2) * 2] * 40 * 30 = 7200 \text{ a/mm}$				Totale: $50 * 8 = 400 \text{ a/mm}$			

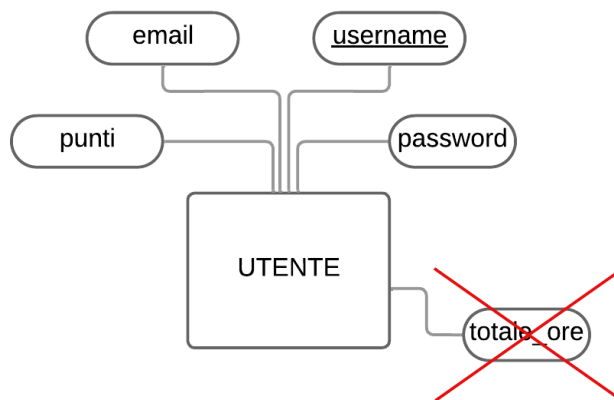
Analisi delle operazioni senza ridondanza

Operazione 2				Operazione 15			
Tabella	Tipo	Accessi	Tipo ac.	Tabella	Tipo	Accessi	Tipo ac.
Gioca	R	1	L	Gioca	R	250	L
Gioca	R	1	S				
Totale: $[1 + (1) * 2] * 40 * 30 = 3600 \text{ a/mm}$				Totale: $250 * 8 = 2000 \text{ a/mm}$			

Totale accessi con ridondanza = $7200 + 400 = 7600 \text{ a/mm} + 400 \text{ byte}$

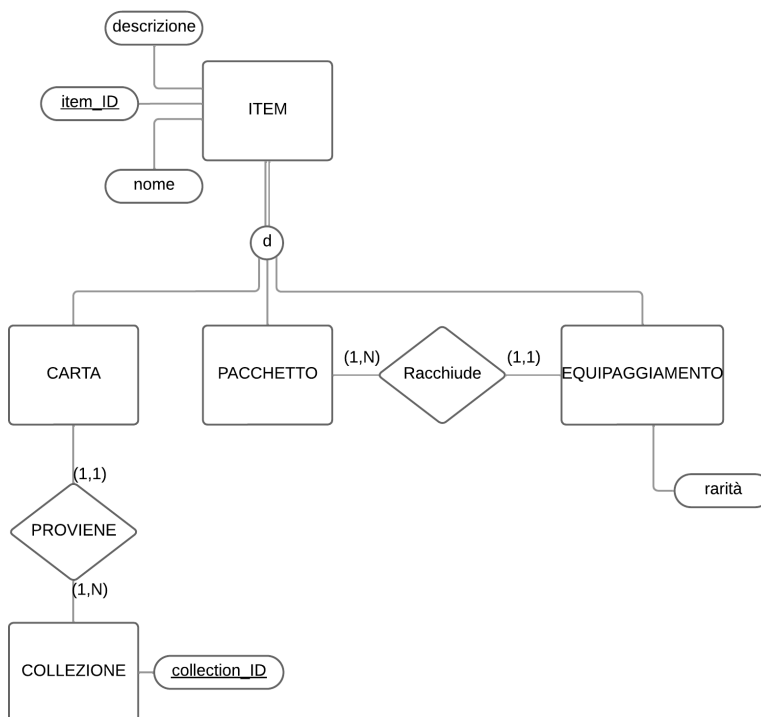
Totale accessi senza ridondanza = $3600 + 2000 = 5600 \text{ a/mm}$

Considerando un numero di accessi maggiore e un'occupazione di memoria di 400 byte, è preferibile non conservare l'attributo ridondante "totale_ore"



Eliminazione delle gerarchie

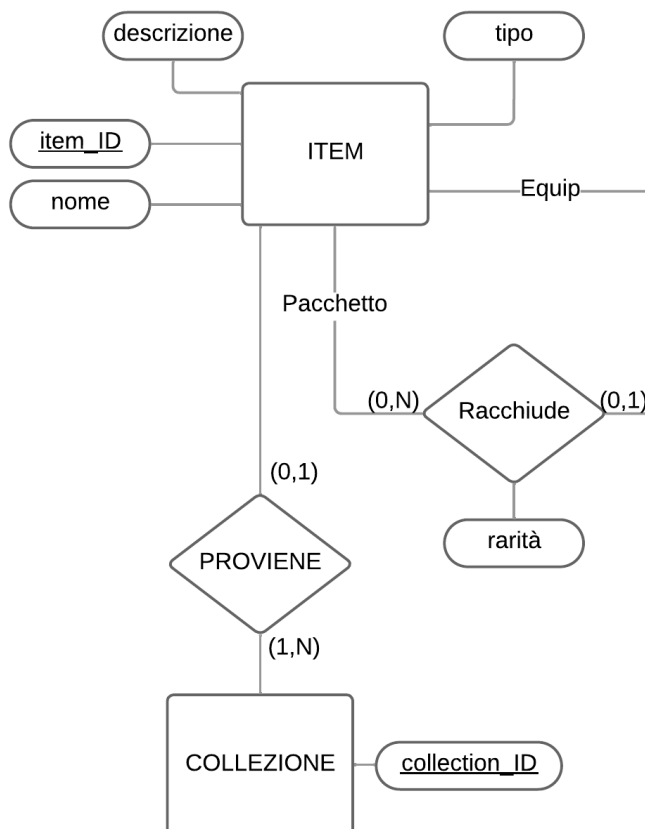
Nella rappresentazione del database iniziale, l'entità "Item" presenta una specializzazione del seguente tipo:



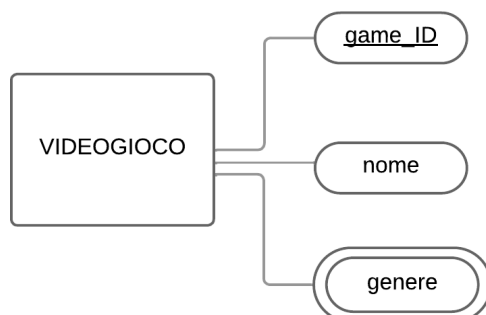
Data la natura dello schema, si è deciso di accorpare le entità "Carta", "Pacchetto" ed "Equipaggiamento" nell'entità padre "Item", che ne erediterà i vari attributi. Per distinguere i

vari tipi di oggetti, è stato aggiunto un attributo “tipo” e la relazione tra “Pacchetto” ed “Equipaggiamento” è stata riproposta come una relazione ricorsiva in Item, su cui sono stati riportati i ruoli che l'entità ricopre in essa.

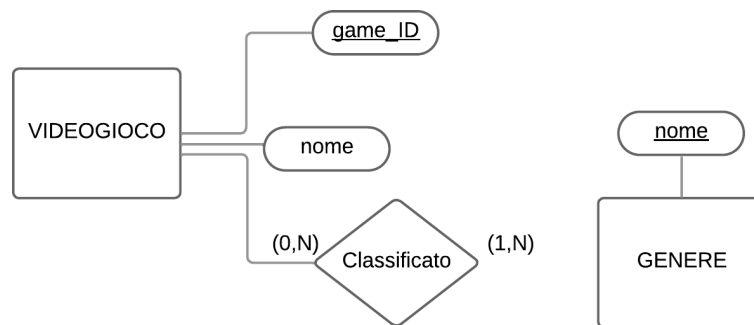
Questo tipo di ristrutturazione andrà a creare dei valori NULL, ma ridurrà significativamente il numero degli accessi delle operazioni.



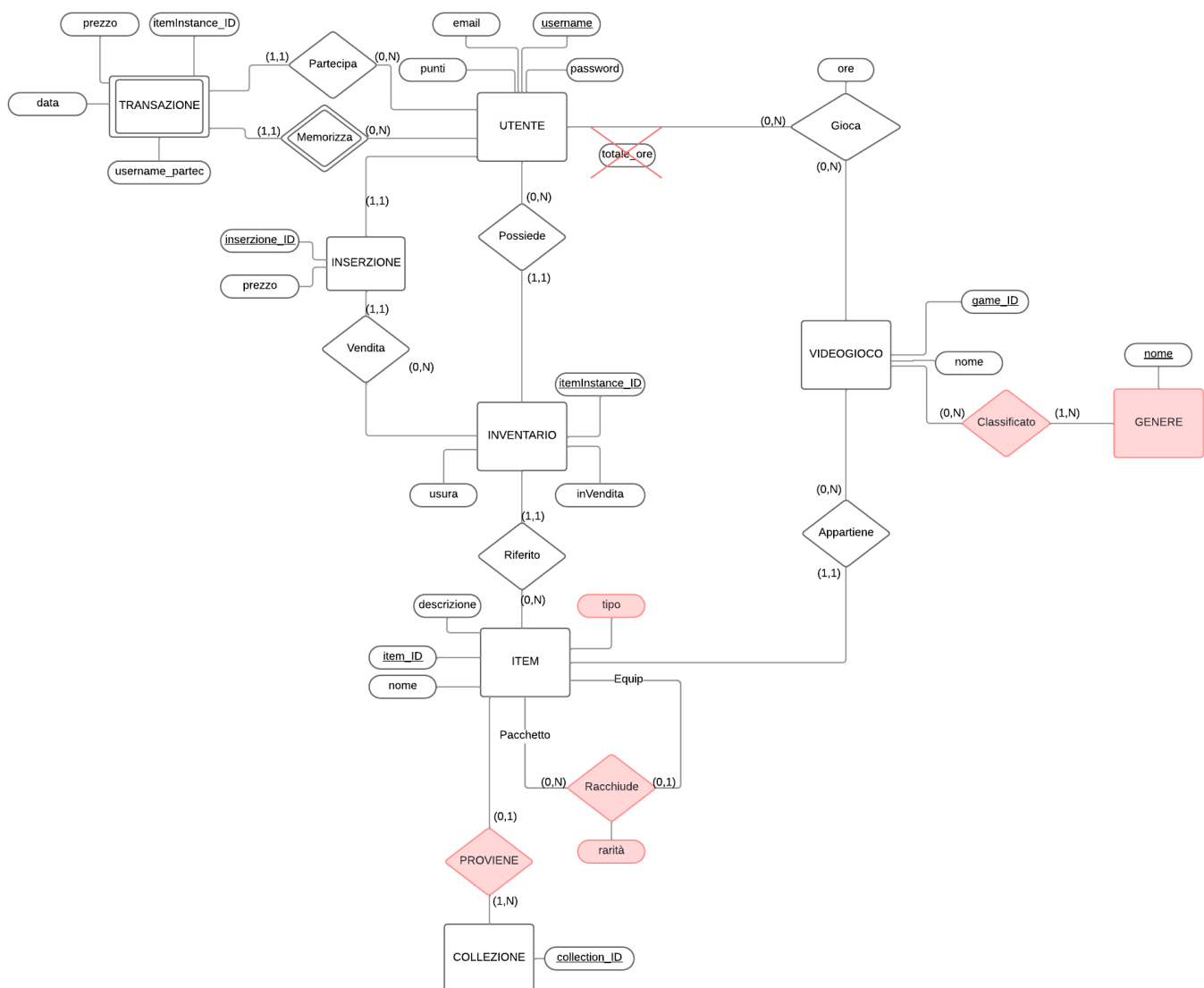
Eliminazione dell'attributo multivalore



L'entità “Videogioco” presenta un attributo multivalore. Si è deciso di sostituire questo attributo con una relazione “Classificato” tra “Videogioco” e una nuova entità “Genere”.



Schema EER Ristrutturato



Schema relazionale

Mapping del database:

User(username, email, password, punti)

Transazione(user.username↑, data_ora, prezzo, itemInstance_ID, username_partec↑)

Videogioco(game_ID, nome)

Genere(nome)

Classificazione(genere.nome↑, videogioco.game_ID↑)

Gioca(user.username↑, videogioco.game_ID↑, totale_ore)

Item(item_ID, nome, descrizione, videogioco.game_ID↑)

Inventario(itemInstance_ID, usura, inVendita, item.item_ID↑, user.username↑)

Collezione(collection_ID)

Proviene(collection_ID↑, item_ID↑)

Racchiude(pacchetto.item_ID↑, equip.item_ID↑, rarità)

Inserzione(inserzione_ID, prezzo, user.username↑, itemInstance_ID↑)

Normalizzazione

Analizziamo il database per effettuare una eventuale normalizzazione.

La base di dati si presenta in prima forma normale in quanto è stato eliminato l'attributo multivalore e tutti i restanti attributi sono atomici.

Tutte le dipendenze funzionali che riguardano chiavi primarie composte da più attributi sono piene e non parziali. Inoltre essendo già in 1NF possiamo dire che è anche in seconda forma normale.

Lo schema è anche in terza forma normale poiché, oltre ad essere in 2NF, ogni attributo non chiave dipende solo dalla chiave primaria e non ci sono dipendenze transitive.

Realizzazione della base di dati con MySQL

Istruzioni per la creazione della base di dati:

```

1 • drop database if exists MyInventory;
2 • create schema MyInventory;
3 • use MyInventory;
4
5 • create table UserAccount
6 (
7     username varchar(50) PRIMARY KEY NOT NULL,
8     email varchar(50) UNIQUE NOT NULL,
9     passwd varchar(50) NOT NULL,
10    punti INT
11 );
12
13 • create table Transazione
14 (
15     username varchar(50),
16     data_ora DATETIME,
17     prezzo INT NOT NULL,
18     itemInstance_ID varchar(50) NOT NULL,
19     username_partec varchar(50) NOT NULL,
20     PRIMARY KEY (username, data_ora),
21     FOREIGN KEY (username) REFERENCES UserAccount(username) ON UPDATE cascade ON DELETE cascade,
22     FOREIGN KEY (username_partec) REFERENCES UserAccount(username) ON UPDATE cascade ON DELETE cascade
23 );
24
25 • create table Genere
26 (
27     nome varchar(50) PRIMARY KEY NOT NULL
28 );
29
30 • create table Videogioco
31 (
32     game_ID int (50) PRIMARY KEY NOT NULL AUTO_INCREMENT,
33     nome varchar(50) NOT NULL
34 ) AUTO_INCREMENT=1;
35
36 • create table Classificazione
37 (
38     game_ID int NOT NULL,
39     genere varchar(50) NOT NULL,
40     PRIMARY KEY(game_ID, genere),
41     FOREIGN KEY (game_ID) REFERENCES Videogioco(game_ID) ON UPDATE cascade ON DELETE cascade,
42     FOREIGN KEY (genere) REFERENCES Genere(nome) ON UPDATE cascade ON DELETE cascade
43 );
44
45 • create table Item
46 (
47     item_ID int PRIMARY KEY NOT NULL AUTO_INCREMENT,
48     nome varchar(50) NOT NULL,
49     descrizione varchar(50) NOT NULL,
50     game_ID int NOT NULL,
51     tipo ENUM('carta', 'pacchetto', 'equip'),
52     FOREIGN KEY(game_ID) REFERENCES Videogioco(game_ID) ON UPDATE cascade ON DELETE cascade
53 ) AUTO_INCREMENT=1;
54
55 • create table ItemInstance
56 (
57     itemInstance_ID int PRIMARY KEY NOT NULL AUTO_INCREMENT,
58     usura float(10,9),
59     invendita bool DEFAULT false,
60     item_ID int,
61     username varchar(50),
62     FOREIGN KEY(username) REFERENCES UserAccount(username) ON UPDATE cascade ON DELETE cascade,
63     FOREIGN KEY(item_ID) REFERENCES Item(item_ID) ON UPDATE cascade ON DELETE cascade
64 ) AUTO_INCREMENT=1;
65

```



```

66 • create table Inserzione
67 (
68     inserzione_ID int PRIMARY KEY NOT NULL AUTO_INCREMENT,
69     prezzo float(6,2),
70     username varchar(50),
71     itemInstance_ID int,
72     FOREIGN KEY(username) REFERENCES UserAccount(username) ON UPDATE cascade ON DELETE cascade,
73     FOREIGN KEY(itemInstance_ID) REFERENCES ItemInstance(itemInstance_ID) ON UPDATE cascade ON DELETE cascade
74 ) AUTO_INCREMENT=1;
75
76 • create table Collection
77 (
78     collection_ID varchar(50) PRIMARY KEY NOT NULL,
79     nome varchar(50) NOT NULL
80 );
81
82 • create table Proviene
83 (
84     carta_ID int NOT NULL,
85     collection_ID varchar(50) NOT NULL,
86     PRIMARY KEY(carta_ID, collection_ID),
87     FOREIGN KEY(carta_ID) REFERENCES Item(item_ID) ON UPDATE cascade ON DELETE cascade,
88     FOREIGN KEY(collection_ID) REFERENCES Collection(collection_ID) ON UPDATE cascade ON DELETE cascade
89 );
90
91 • create table Racchiude
92 (
93     pacchetto_ID int,
94     equip_ID int,
95     rarità enum('comune', 'raro', 'epico', 'leggendario'),
96     PRIMARY KEY(pacchetto_ID, equip_ID),
97     FOREIGN KEY(equip_ID) REFERENCES Item(item_ID) ON UPDATE cascade ON DELETE cascade,
98     FOREIGN KEY(pacchetto_ID) REFERENCES Item(item_ID) ON UPDATE cascade ON DELETE cascade
99 );
100
101 • create table Gioca
102 (
103     username varchar(50) NOT NULL,
104     game_ID int NOT NULL,
105     totale_ore INT,
106     PRIMARY KEY(username, game_ID),
107     FOREIGN KEY(username) REFERENCES UserAccount(username) ON UPDATE cascade ON DELETE cascade,
108     FOREIGN KEY(game_ID) REFERENCES Videogioco(game_ID) ON UPDATE cascade ON DELETE cascade
109 );

```

Implementazione query SQL

Operazione 1:

```
INSERT INTO UserAccount (username, email, passwr, punti) VALUES  
(  
    ?, ?, ?, ?  
);
```

Operazione 2:

```
INSERT INTO Gioca(username, game_ID, totale_ore) VALUES  
(  
    ?, ?, ?  
) ON DUPLICATE KEY UPDATE totale_ore = totale_ore + ?;
```

```
UPDATE UserAccount  
SET punti = punti + (10 * n)  
WHERE username = ?;
```

Operazione 3:

```
INSERT INTO Videogioco (nome) VALUES  
(  
    ?  
);
```

Operazione 4:

```
INSERT INTO Inserzione(prezzo, username, itemInstance_ID) VALUES
(
    ?, ?, ?
);
```

```
UPDATE ItemInstance
SET inVendita = true
WHERE itemInstance_ID = ?;
```

Operazione 5:

```
INSERT INTO ItemInstance (usura, item_ID, username)
SELECT RAND(), racchiude.equip_ID, '?'           -- ? = username dell'utente che
effettua l'operazione
FROM Racchiude
WHERE pacchetto_ID = ?                           -- ? = item_ID del pacchetto aperto
ORDER BY RAND()                                 -- Ordina casualmente i risultati
LIMIT 1;                                         -- Seleziona solo un equipaggiamento
casuale

-- Eliminazione del pacchetto aperto
DELETE FROM ItemInstance WHERE ItemInstance_ID = ?; -- ? = ItemInstance_ID del
pacchetto aperto
```

Operazione 6:

```
SELECT nome, UserAccount.username, ItemInstance_ID
FROM ItemInstance, Item, UserAccount
WHERE ItemInstance.username=UserAccount.username and
Item.item_ID=ItemInstance.item_ID
ORDER BY UserAccount.username;
```

Operazione 7:

```
INSERT INTO ItemInstance (usura, item_ID, username)
SELECT NULL, item_ID, '?'                                -- ? = username dell'utente che effettua
l'operazione
FROM item
WHERE item.tipo = 'pacchetto'                             -- ? = item_ID del pacchetto aperto
ORDER BY RAND()                                           -- Ordina casualmente i risultati
LIMIT 1;                                                  -- Seleziona solo un equipaggiamento
casuale

-- Eliminazione delle carte selezionate
DELETE FROM ItemInstance
WHERE ItemInstance_ID = ? OR ItemInstance_ID = ? OR ItemInstance_ID = ? OR
ItemInstance_ID = ? OR ItemInstance_ID = ?
```

Operazione 8:

```
INSERT INTO ItemInstance (usura, item_ID, username)
SELECT RAND(), item_ID, '?'                                -- ? = username dell'utente che effettua
l'operazione
```

```
FROM item
WHERE item.tipo = 'carta'                -- ? = item_ID del pacchetto aperto
ORDER BY RAND();
```

```
UPDATE UserAccount
SET punti = punti-10
WHERE username = ?;
```

Operazione 9:

```
SELECT nome, UserAccount.username, ItemInstance_ID, rarità
FROM ItemInstance, Item, UserAccount, Racchiude
WHERE ItemInstance.username= '?' AND Racchiude.rarità = 'legendario' AND Item.item_ID
= ItemInstance.item_ID AND UserAccount.username = ItemInstance.username AND
Racchiude.equip_ID = Item.item_ID
ORDER BY UserAccount.username;
```

Operazione 10:

```
SELECT username
FROM Gioca
GROUP BY username
HAVING COUNT(DISTINCT game_ID) >= 5;
```

Operazione 11:

```
INSERT INTO Item(nome, descrizione, game_ID, tipo) VALUES
( ?, ?, ?, 'pacchetto);
```



```
INSERT INTO Racchiude (pacchetto_ID, equip_ID, rarità) VALUES  
( ?, ?, ? );
```

Operazione 12:

```
INSERT INTO Racchiude (pacchetto_ID, equip_ID, rarità) VALUES  
( ?, ?, ? );
```

Operazione 13:

```
SELECT nome, prezzo  
FROM Inserzione, ItemInstance, Item  
WHERE Inserzione.ItemInstance_ID = ItemInstance.itemInstance_ID and  
ItemInstance.item_ID = Item.Item_ID  
and nome='?';
```

Operazione 14:

```
SELECT UA.username, COUNT(GC.game_ID) AS num_videogiochi_giocati  
FROM UserAccount UA  
LEFT JOIN Gioca GC ON UA.username = GC.username  
GROUP BY UA.username  
ORDER BY num_videogiochi_giocati DESC;
```

Operazione 15:

```
SELECT UA.username, SUM(g.totale_ore) AS ore_totali
FROM UserAccount UA
LEFT JOIN Gioca g ON UA.username = G.username
GROUP BY UA.username
ORDER BY ore_totali DESC;
```

Operazione 16:

```
SELECT username, COUNT(*) AS num_items
FROM ItemInstance
GROUP BY username
ORDER BY num_items DESC
LIMIT 1;
```

Operazione 17:

```
DELETE FROM Inserzione
WHERE inserzione_ID = ' ? ';
```

```
UPDATE ItemInstance
SET username = ' ? '
WHERE itemInstance_ID = ' ? ';
```

```
INSERT INTO Transazione (username, data_ora, prezzo, itemInstance_ID, username_partec)
VALUES ( ' ? ', NOW(), ' ? ', ' ? ', ' ? ' );
```

Operazione 18:

```
SELECT username
FROM UserAccount
WHERE (
    SELECT COUNT(*)
    FROM ItemInstance
    WHERE ItemInstance.username = UserAccount.username
) >= 5
AND EXISTS (
    SELECT *
    FROM Transazione
    WHERE Transazione.username = UserAccount.username
);
```

Operazione 19:

```
SELECT II.*
FROM ItemInstance II
WHERE II.username IN (
    SELECT G.username
    FROM Gioca G
    WHERE G.game_ID = (
        SELECT game_ID
        FROM Videogioco
        WHERE nome = 'Dota 2'
    )
) AND II.item_ID IN (
    SELECT R.equip_ID
    FROM Racchiude R
    JOIN Item I ON R.equip_ID = I.item_ID
    WHERE R.rarità = 'leggendario'
);
```