

A dark blue vertical bar runs down the left side of the page. A light blue arrow points to the right from the bar, containing the text 'N. 32'.

N. 32

# Digital Delivery & Shipping

Several thin, curved lines in shades of blue and grey originate from the bottom left and sweep upwards and to the right.

Algeri Riccardo, Cavalli Mattia  
e Stigliano Lorenzo

# Indice

## 0. Introduzione

## 1. Informazioni Generali

## 2. Progettazione Concettuale

### 2.1 Glossario

### 2.2 Dallo Schema Scheletro allo Schema ER

### 2.3 Analisi Entità

### 2.4 Analisi Associazioni

## 3. Progettazione Logica

### 3.1 Eliminazione Gerarchie Isa

### 3.2 Selezione chiavi primarie ed eliminazione id esterne

### 3.3 Traduzione di entità e associazioni in schemi relazionali

### 3.4 Normalizzazione

## 4. Dato Derivato

## 5. Progetto Fisico

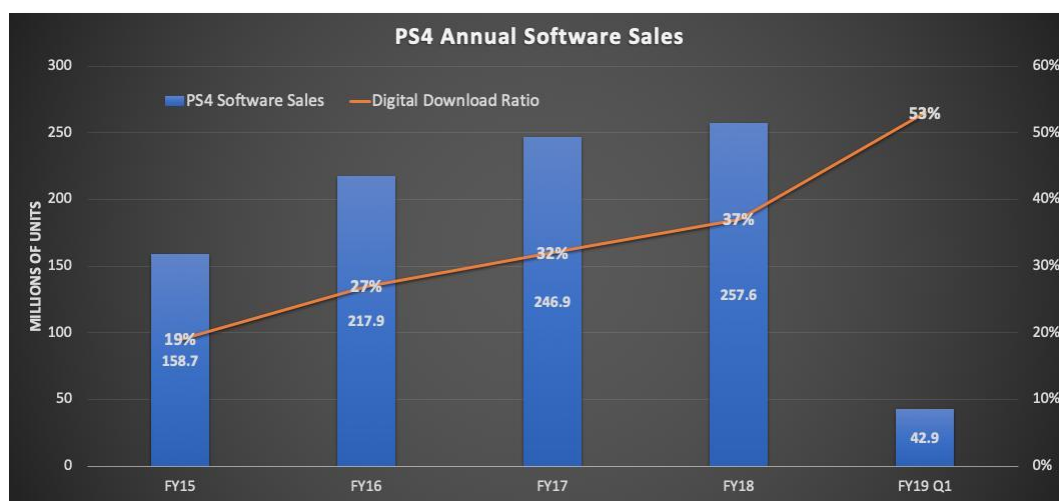
## 6. Table, Trigger e Dati

## 7. Interrogazioni DBMS

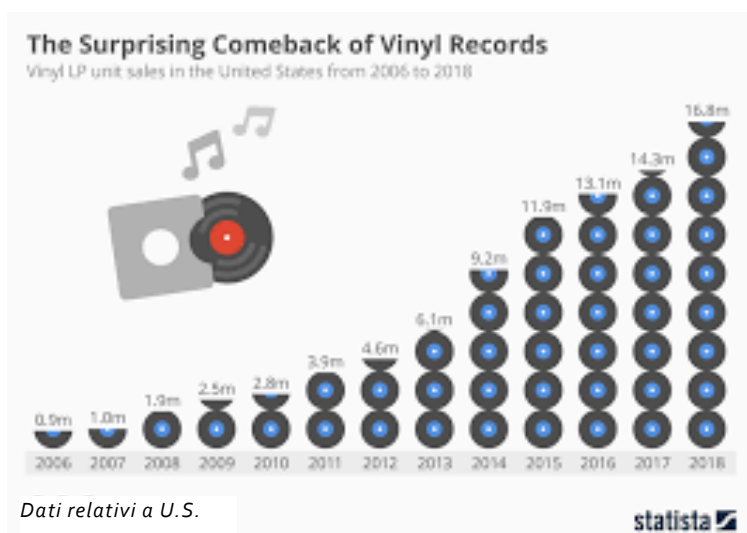
## 8. JDBC

## 0. Introduzione

Digital Delivery & Shipping è un progetto che si pone l'obiettivo di dare agli utenti la possibilità di scegliere il formato del videogioco acquistato. Lo abbiamo creato avendo in mente una fetta di pubblico che preferisce le copie fisiche (soprattutto per le edizioni limitate o da collezione) quando acquista videogiochi. L'analista videoludico Daniel Ahmad afferma che nell'anno fiscale 2019 sulla piattaforma PS4, il 53% dei videogiochi è stato venduto in formato digitale. Un dato importante, se si pensa che nel 2015 la percentuale era solamente del 19%.



Questo avrebbe dovuto spingerci a desistere dall'intraprendere questo percorso, apparentemente giunto all'epilogo, ma abbiamo voluto immaginare più in grande.



Fino al 2006 tutti pensavano che il mercato degli LP, i famosi dischi in vinile da 33 giri, fosse morto. Quello che accadde dal 2007 in poi fu una vera e propria crescita esponenziale che ha portato le vendite annue dei vinili dal minimo storico di 5 milioni copie a circa 45 milioni, nel 2018. Come è stato

possibile tutto ciò? Perché le persone si sono interessate ad una tecnologia così lontana dal comodissimo streaming? La nostalgia. Le persone vivono di ricordi, sono appassionate al passato, più che al futuro ed in questo senso immaginiamo che il nostro DBMS possa aiutare un di E-commerce che si occupi di questa fetta di mercato ad oggi in calo, ma che sicuramente tornerà prepotentemente e che già oggi fa sentire il suo arrivo l'onda del Retro Gaming sta Arrivando!



# 1. Informazioni Generali

Il DBMS si occupa della gestione de dati relativi ad un sito di E-Commerce che come dicevamo lavora nel campo del gaming. Questo ha necessità di gestire utenti, videogiochi e ordini da parte degli utenti, nonché fungere da HUB per i giocatori, i quali possono scambiarsi l'amicizia fra di loro.

Ciascuna delle entità che andremo a presentare ha un suo ID di riconoscimento e talvolta questo è composto da un chiave interna ed una esterna.

Il DBMS si occupa anche della gestione di un piccolo magazzino che contiene le copie fisiche dei videogiochi. Questo aggiunge dettaglio all'entità videogioco, che oltre ad essere ricca di attributi propri ed essere collegata ad una categoria, possiede anche una posizione concreta all'interno del magazzino. Non tutti i giochi saranno in copia fisica e questi saranno acquistabili solamente in copia digitale.

Elenchiamo di seguito alcune le funzioni base del sito:

- Mostrare i videogiochi all'utente
- Permettere all'utente di cercare videogiochi grazie al nome o alcuni degli attributi di questi ultimi
- Permettere al cliente di acquistare i videogiochi presenti sul sito, che in seguito verranno aggiunti alla libreria personale
- Interazioni con altri utenti grazie alle richieste di amicizia

Flusso standard:

- Atterrare nella home page (questo aspetto è relegato alla programmazione del sito web) dove verranno presentati i prodotti che si vogliono mettere in evidenza
- Accedere alle categorie di prodotti
- Accedere alle schede di dettaglio di prodotti
- Inserire i videogiochi desiderati nell'ordine e procedere all'acquisto
- Se richiesta copia fisica, verificarne la presenza sui ripiani degli scaffali presenti in magazzino
- Inviare una richiesta ad un amico

## 2. Progettazione Concettuale

### 2.1 Glossario

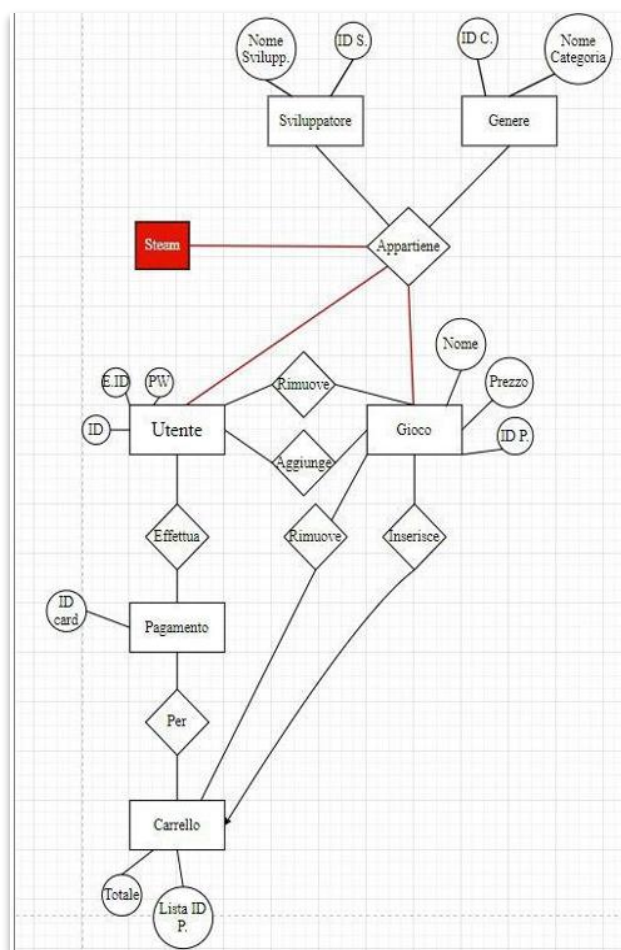
A seguito delle valutazioni di carattere generale che abbiamo esposto prima, ci siamo concentrati fin da subito sulla creazione di un glossario completo, chiaro e semplice, che avrebbe potuto aiutarci nella creazione di un robusto schema ER.

Termine	Descrizione	Sinonimi	Collegamenti
Videogioco	Oggetto acquistabile in copia digitale e fisica	Videogame, Gioco, Prodotto	Ordine, Categoria, Ripiano, Libreria, Recensione
Utente	Persona registrata al sito che può effettuare acquisti; avente una libreria contenente i titoli acquisti e con la possibilità di stringere amicizia con altri utenti	User, Giocatore	Ordine, Libreria, Recensione
Amico	Utente amico di uno o più utenti		Utente
Categoria	Categoria che in cui ricade un determinato videogioco		Videogioco
Ordine	Insieme di tutti i videogiochi che sono pronti per essere acquistati		Videogioco, Utente, Pagamento
Pagamento	Pagamento di un determinato ordine		Ordine
Carta di Credito	Un particolare metodo di pagamento		Pagamento
PayPal	Un particolare metodo di pagamento		Pagamento
Libreria	Lista dei giochi acquistati, legata ad un determinato utente		Utente
Spedizione	Entità relativa alla spedizione di un videogioco di cui è stata richiesta la copia fisica		Ordine
Ripiano	Locazione all'interno di un magazzino di un determinato titolo		Videogioco, Magazzino
Magazzino	Magazzino contenente videogiochi		Ripiano

Recensione	Commento legato ad un determinato videogioco composto da un utente		Videogioco, Utente
------------	--	--	--------------------

## 2.2 Dallo Schema Scheletro allo Schema ER

Glossario alla mano abbiamo iniziato a disegnare lo schema scheletro (release 0.8):



Release 0.8

siamo partiti dalle entità che ritenevamo fossero più importanti: Utente, Gioco e componendo le relazioni fondamentali. Non sapevamo come gestire la ridondanza delle associazioni e dei dati, ma avevamo in mente quali fossero gli obiettivi del DBMS: poter cercare un videogioco, inserirlo nell'ordine, procedere all'acquisto.

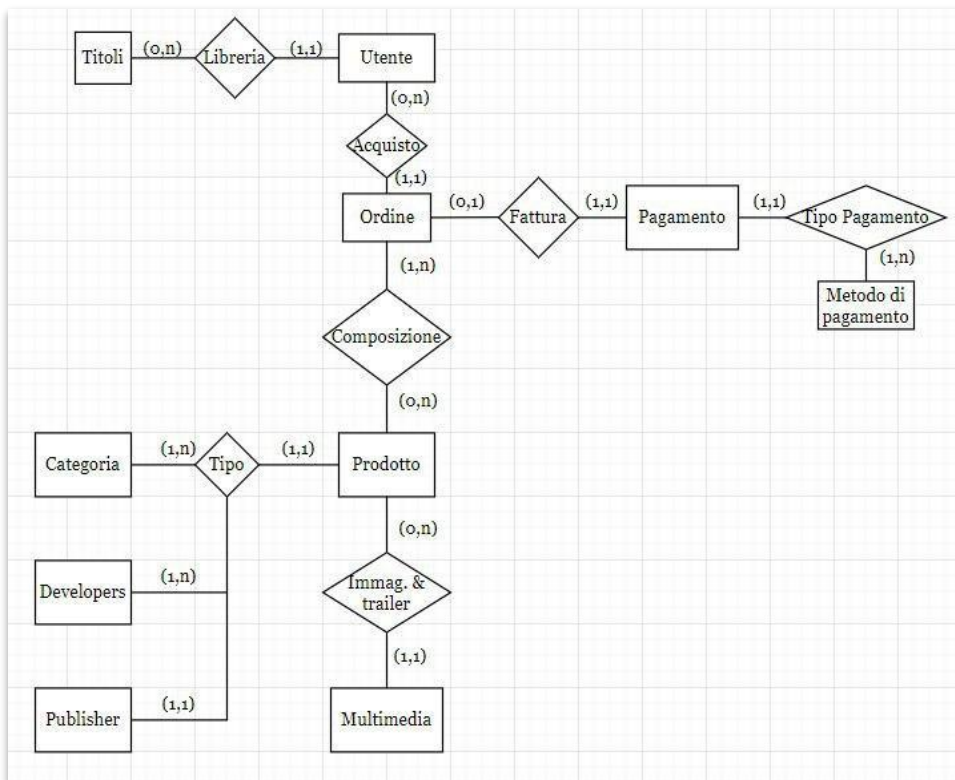
Abbiamo così generato un secondo schema (0.9) e poi lo abbiamo arricchito con il suo seguito (release 0.9.1).

Nonostante gli enormi passi in avanti non avevamo ancora risolto quelli che erano problemi come:

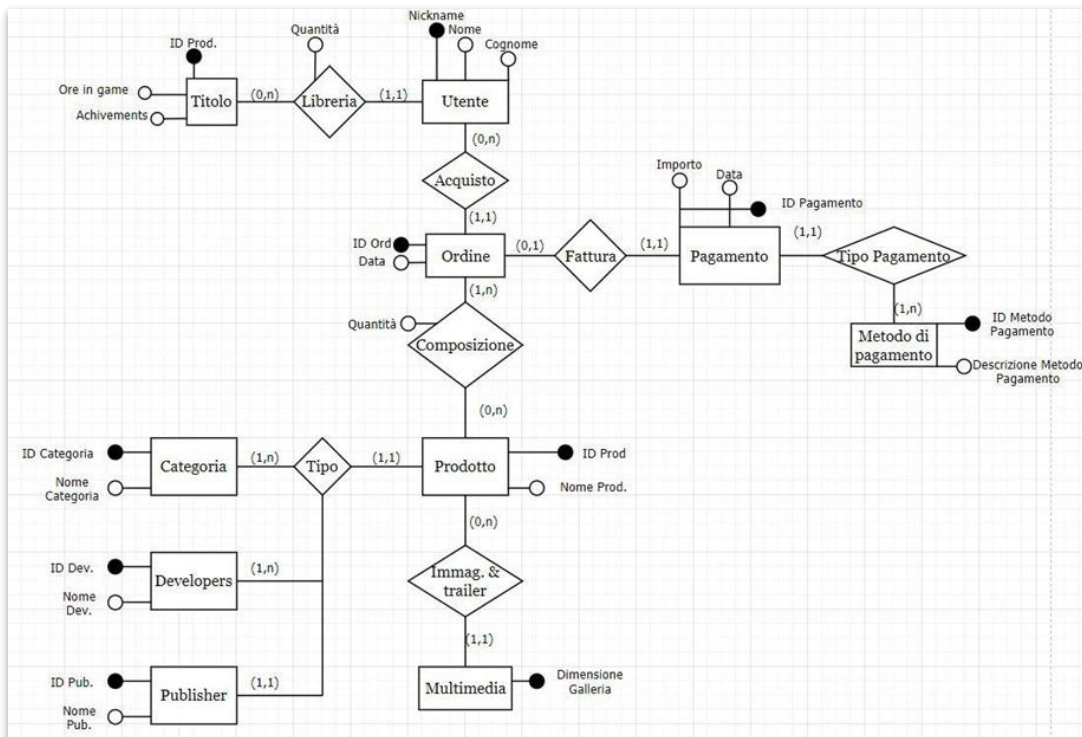
- Ridondanza
- Associazioni concettualmente errate
- Strutturalmente fragile
- Povero di dettaglio



Continuando però a progettare e correggere di volta in volta il nostro DBMS siamo poi arrivati alla sua versione finale.

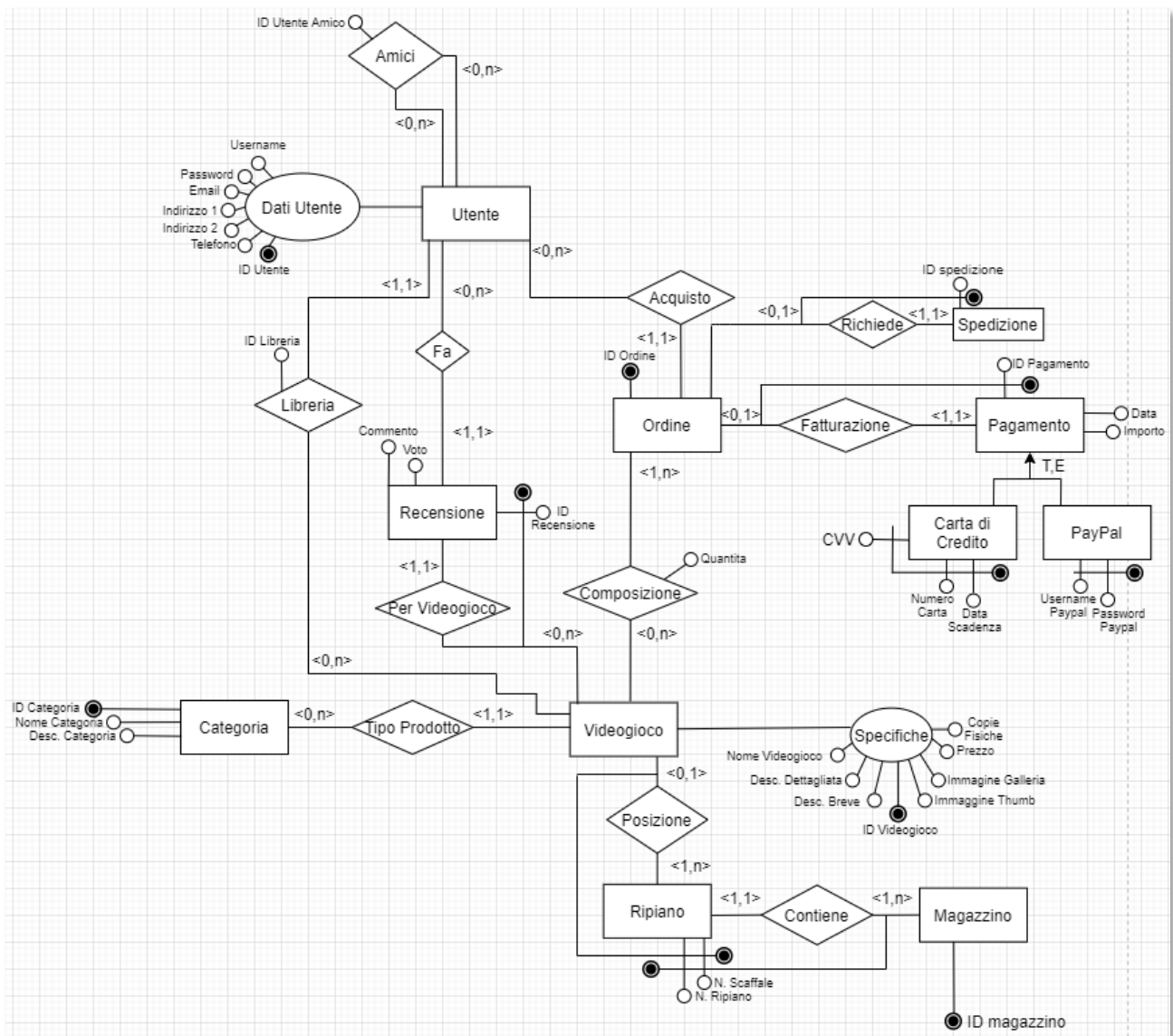


Release 0.9



Release 0.9.1





L'utente ora ha una libreria, può formulare ordini e può inviare richieste di amicizie agli amici.

Gli ordini possono essere pagati con due metodi di pagamento diversi ed è possibile richiedere la spedizione, qualora un utente desiderasse ricevere la copia fisica del proprio titolo preferito.

La gestione del videogioco è fatta su due piani differenti: uno digitale attraverso le categorie, ed uno fisico attraverso ripiano e magazzino. Un titolo quindi può essere presente in magazzino oppure essere solamente digital delivery only.

I videogiochi possono ricevere recensioni con voto, ma ogni utente può formulare unicamente 1 recensione per singolo gioco.

## 2.3 Analisi delle entità

Utente	
IDUtente	È il codice univoco di 5 cifre (char) che viene associato all'utente nel momento della registrazione al sito. Può identificare qualsiasi cliente del sito ed è la chiave primaria dell'entità cliente.
Username	È lo pseudonimo dell'utente per accedere al sito, è unico e non deve essere nullo (massimo 30 caratteri).
Password	È la password che l'utente deve inserire per accedere al proprio account sul sito, deve essere non nulla (massimo 30 caratteri).
Email	Indirizzo email dell'utente.
Indirizzo1	Indirizzo di spedizione primario (massimo 60 caratteri)
Indirizzo2	Indirizzo di spedizione secondario (massimo 60 caratteri)
Telefono	Numero di telefono dell'utente per l'autoidentificazione a due fattori o il recupero delle credenziali (massimo 10 caratteri numerici).

Ordine	
ID_Ordine	È il codice univoco di 5 caratteri (char) che identifica l'ordine ed è la chiave primaria di tale entità.

Videogioco	
ID_Videogioco	È il codice univoco di 5 cifre (char) che identifica il videogioco ed è la chiave primaria di tale entità.
Nome Videogioco	È il nome del videogioco formato da 30 caratteri (varchar), non null e unico.
Descrizione Breve	È la descrizione breve che verrà visualizzata in anteprima del prodotto, fatta da 200 caratteri alfanumerici.
Descrizione Dettagliata	È la descrizione dettagliata che verrà visualizzato nella scheda del prodotto, fatta da 1000 caratteri alfanumerici.

Immagine Thumb Videogioco	È l'immagine del prodotto che verrà visualizzata nella pagina di categoria. Abbiamo deciso di rappresentarla con un Booleano per segnalarne la presenza o meno.
Immagine full Videogioco	È l'immagine del prodotto che verrà visualizzata nella scheda del videogioco. Abbiamo deciso di rappresentarla con un Booleano per segnalarne la presenza o meno.
Prezzo	È il prezzo di vendita (Integer) del videogioco, che deve essere > 0
Copie_Fisiche	Numero di copie fisiche esistenti.

Recensione	
IDRecensione	È il codice univoco di 5 cifre (char) che identifica la recensione ed è la chiave primaria di tale entità combinata con IDUtente.
Voto	Voto (smallint) della recensione
Commento	Commento (varchar) di 500 caratteri che rappresenta la recensione scritta.
ID_Utente	ID_Utente che serve per formulare la primary key dell'entità Recensione: PRIMARY KEY (ID_RECENSIONE, ID_UTENTE, ID_VIDEOGIOCO). <u>FK</u> → Utente
ID_Videogioco	ID_Videogioco che serve per formulare la primary key dell'entità Recensione: PRIMARY KEY (ID_RECENSIONE, ID_UTENTE, ID_VIDEOGIOCO). <u>FK</u> → Videogioco

Categoria	
ID_Categoria	È il codice univoco (char) di 5 cifre che identifica la categoria ed è la chiave primaria dell'entità.
Nome Categoria	Nome della categoria formato da 30 caratteri (varchar).
Descrizione Categoria	Descrizione della categoria formata da 300 caratteri (varchar).

Pagamento	
ID_Pagamento	È il codice (char) univoco di 5 cifre associato al pagamento è la chiave primaria per tale entità combinata con ID_Ordine.
Data	È la data (date) in cui è stato effettuato il pagamento.
Importo	Importo totale numeric (6,2).

Carta di Credito	
IDPagamento	È il codice (char) univoco di 5 cifre associato al pagamento è la chiave primaria per tale entità combinata con ID_Ordine. PRIMARY KEY (ID_ORDINE, ID_PAGAMENTO). FK → Ordine
Numero Carta	Numero carta di credito di 16 caratteri (char).
Data Scadenza	Data scadenza carta di credito (date).
CCV	Numero segreto di 3 cifre (numeric) posto sul retro della carta di credito.


PayPal	
IDPagamento	È il codice di 5 cifre (char) univoco associato al PayPal. PRIMARY KEY (ID_PAGAMENTO, USERNAME PAYPAL, PASSWORD PAYPAL). FK → PAGAMENTO
Username PayPal	Username (varchar) di massimo 30 caratteri.
Password PayPal	Password (varchar) di massimo 30 caratteri.

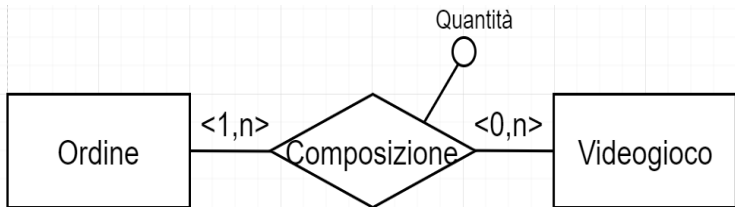
Spedizione	
ID_Spedizione	È il codice (char) di 5 cifre associato al pagamento ed è la chiave primaria per tale entità.
ID_Ordine	È il codice univoco di 5 caratteri (char) che identifica l'ordine ed è la chiave primaria di tale entità.

Magazzino	
IDMagazzino	È il codice (char) univoco di 3 cifre associato al ripiano ed è la chiave primaria per tale entità.


Ripiano	
N_Ripiano	È il codice (char) univoco di 3 cifre associato al ripiano ed è la chiave primaria per tale entità assieme a N_Scaffale e ID Magazzino
N_Scaffale	È il codice (char) univoco di 3 cifre associato all'entità ripiano che identifica un determinato scaffale.
ID Magazzino	È il codice (char) univoco di 3 cifre associato al magazzino. <u>FK</u> → Magazzino


## 2.4 Analisi Associazioni

Acquisto	
Collega l'entità "cliente" con l'entità "ordine" e rappresenta l'immissione di un ordine da parte di un cliente.	
<u>Cardinalità</u> 	Ogni ordine è associato ad un solo cliente, ma ogni cliente può fare più ordini (da 0 a n). La partecipazione dell'entità ordine è obbligatoria in quanto ogni "ordine" deve essere stato immesso da un cliente, mentre la partecipazione di cliente è facoltativa in quanto vi possono essere "clienti" che non fanno ordini.

Composizione	
Collega l'entità "ordine" con l'entità "videogioco" e rappresenta la presenza di determinati prodotti all'interno di un ordine	
<u>Cardinalità</u> 	Ogni ordine può essere composto da 1 o più videogiochi e ogni videogioco può far parte di più ordini. La partecipazione di "ordine" è obbligatoria in quanto ogni ordine deve contenere almeno un videogioco, mentre l'entità "videogioco" è facoltativa in quanto vi possono essere dei

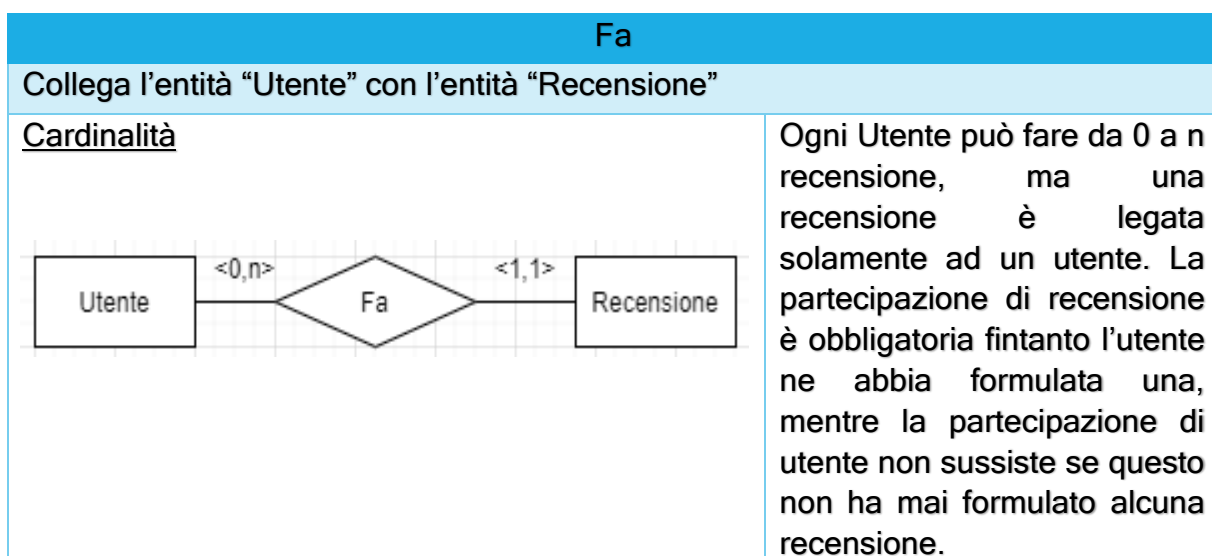
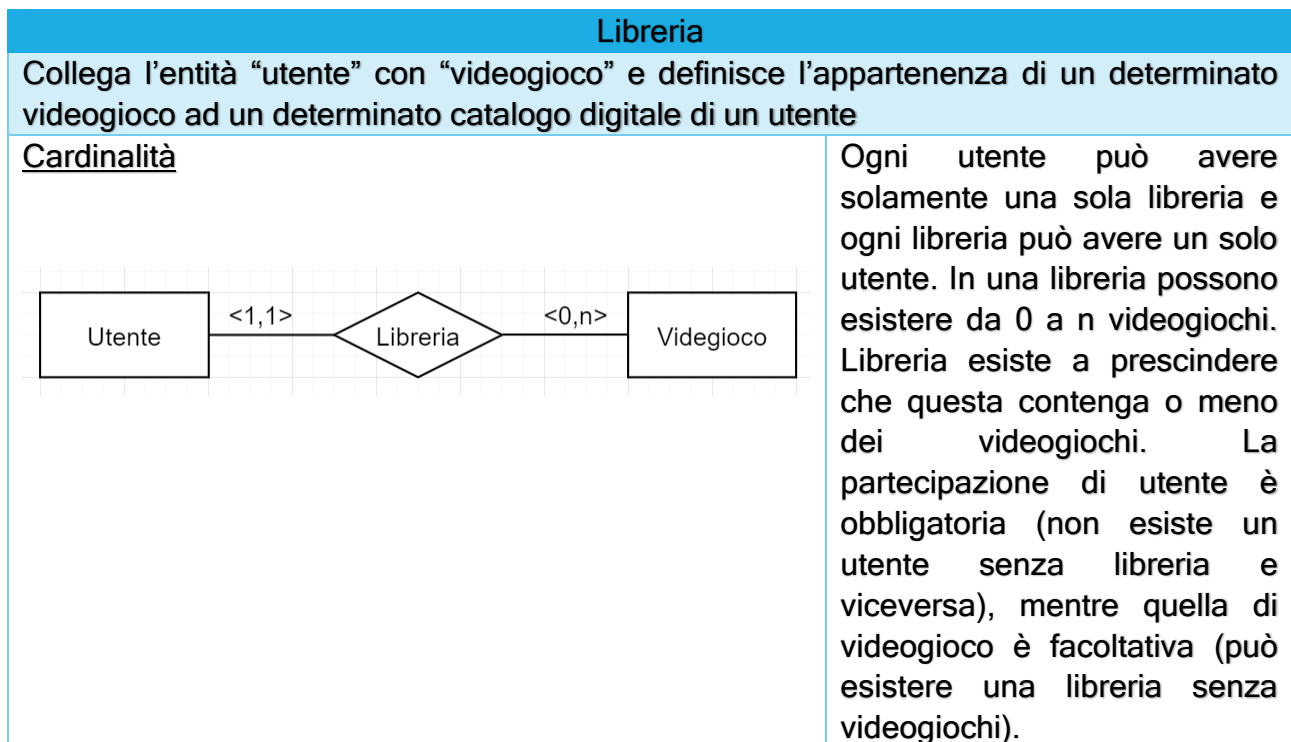
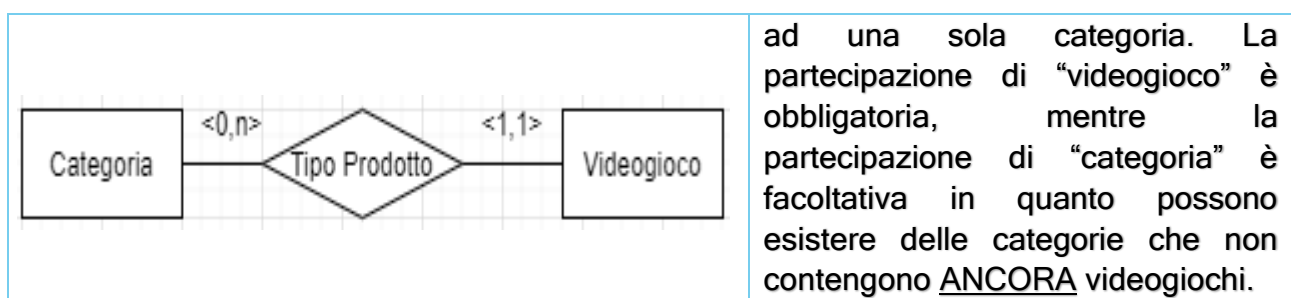
	videogiochi non associati a nessun ordine.
<u>Quantità</u>	È la quantità di un prodotto associata ad un determinato ordine (numeric (2)).

Fatturazione	
Collega l'entità "ordine" con l'entità "pagamento", definendo il pagamento/fatturazione dell'ordine	
<u>Cardinalità</u>  <pre> graph LR     Ordine[Ordine] -- "&lt;0,1&gt;" --- Fatturazione{Fatturazione}     Fatturazione -- "&lt;1,1&gt;" --- Pagamento[Pagamento] </pre>	<p>Ogni ordine può essere pagato una e una sola volta e un pagamento può essere associato a solamente un ordine. La partecipazione di "pagamento" è obbligatoria perché ogni pagamento deve essere associato ad un ordine, ma la partecipazione di "ordine" è facoltativa in quando un ordine può essere stato immesso, ma può non essere ancora stato pagato.</p>

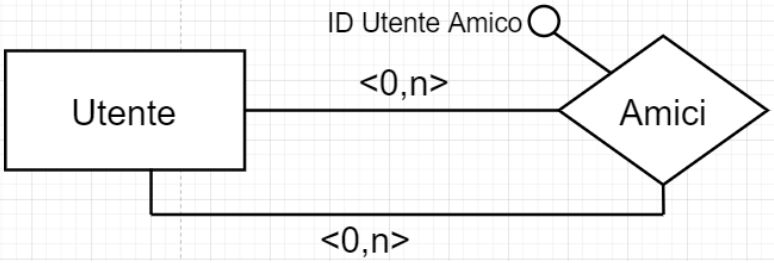
Richiede	
Collega l'entità "ordine" con l'entità "spedizione" e definisce la presenza o meno della spedizione di un ordine	
<u>Cardinalità</u>  <pre> graph LR     Ordine[Ordine] -- "&lt;0,1&gt;" --- Richiede{Richiede}     Richiede -- "&lt;1,1&gt;" --- spedizione[spedizione] </pre>	<p>Ogni ordine può essere richiedere la spedizione o non richiederla affatto. La partecipazione di spedizione è obbligatoria, se possibile, quando l'utente chiede la spedizione di un ordine; anche per quanto riguarda la partecipazione di ordine, questa è obbligatoria solo se l'utente richiede la spedizione.</p>


Tipo di Prodotto	
Collega l'entità "videogioco" con l'entità "categoria" e definisce l'appartenenza di un prodotto ad una determinata categoria.	
<u>Cardinalità</u>	Ogni categoria può avere da 0 a n videogiochi al suo interno mentre un videogioco deve essere associato







Amici	
Collega l'entità "Utente" con l'entità "Utente"	
<p><u>Cardinalità</u></p> 	<p>Ogni Utente può essere amico di altri utenti per un numero di volte che va da 0 a n e allo stesso modo può avere a sua volta per amico altri utenti da 0 a n. Nessuna delle due partecipazioni è obbligatoria.</p>
<p><u>ID Utente Amico</u></p>	<p>Chiave dell'utente con cui l'entità utente è amico. È una variabile char(5), not null.</p>

Per Videogioco	
Collega l'entità "Recensione" con l'entità "Videogioco"	
<p><u>Cardinalità</u></p> 	<p>Ogni videogioco può avere 0 a n recensione, ma un videogioco può avere solamente una recensione per ciascun utente.</p>

## Posizione

Collega l'entità "Videogioco" con l'entità "Ripiano"

### Cardinalità



Ogni videogioco può essere presente o meno su un ripiano e un ripiano, se presente il videogioco, può avere da 1 a n videogiochi dello stesso tipo. La partecipazione di videogioco non è obbligatoria perché possono sussistere titoli senza copia fisica, ma se questi esistono la partecipazione di ripiano è obbligatoria, poiché allora deve esistere su qualche ripiano.

## Contiene

Collega l'entità "Ripiano" con l'entità "Magazzino"

### Cardinalità

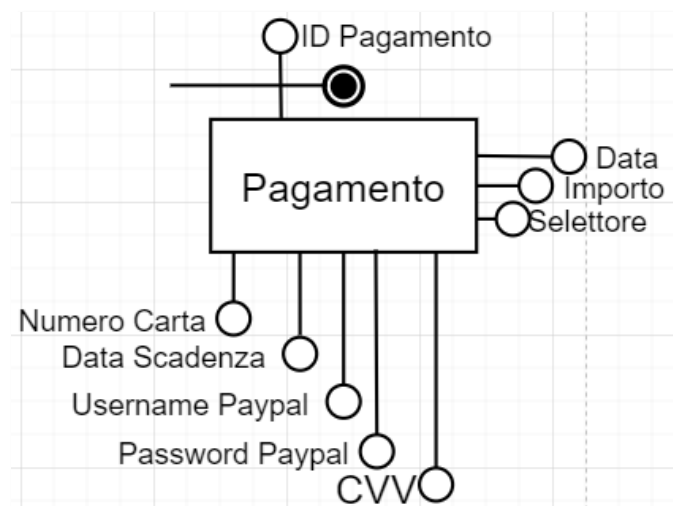


Un determinato ripiano, di un determinato scaffale, è contenuto in un solo magazzino, ma un magazzino ha da 1 a n ripiani di determinati scaffali. La partecipazione di ripiano è obbligatoria, in quanto un magazzino ha almeno un ripiano e anche quella di magazzino è obbligatoria, perché per un ripiano deve esistere un magazzino in cui questo è contenuto.

## 3. Progettazione Logica

### 3.1 Eliminazione Gerarchie

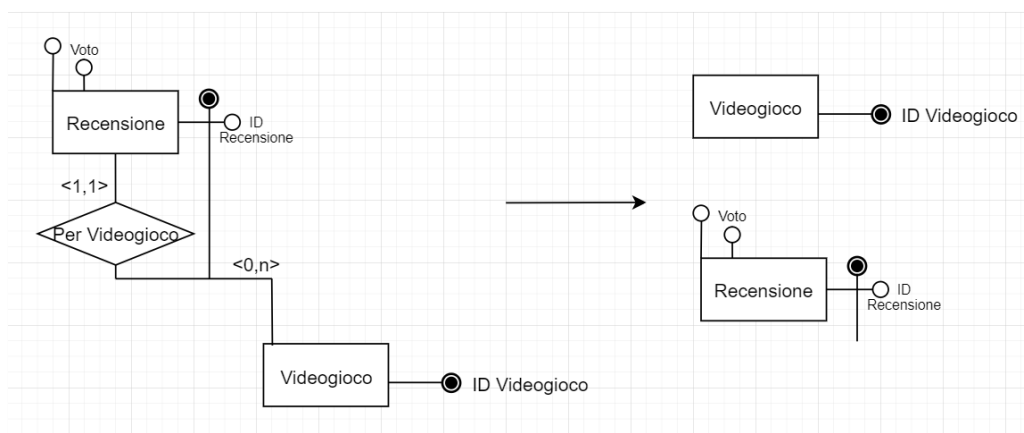
L'unica gerarchia presente all'interno del nostro DBMS è quella che riguarda il pagamento, che ha come entità figlie Carta di credito e PayPal. Per risolverla abbiamo pensato ad un collasso verso l'alto coordinato con l'utilizzo di un selettore ed un trigger. Il selettore per decidere se il pagamento viene effettuato con il metodo PayPal/Carta di Credito ed il trigger per evitare che venga inserito un pagamento con i dati sbagliati rispetto allo stato del selettore.



## 3.2 Selezione chiavi primarie ed Eliminazione ID Esterne

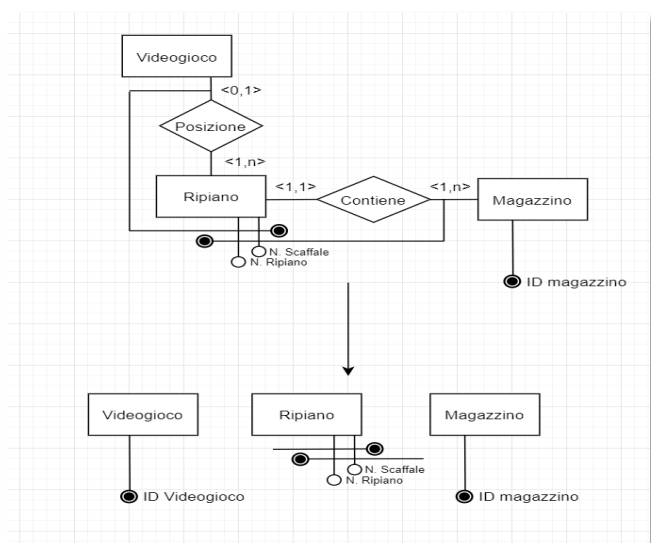
### Videogioco - Recensione

L'entità recensione è identificata da ID\_Recensione e ID\_Videogioco, attraverso l'associazione "Per Videogioco". Come chiave di Recensione si mantiene la chiave composta, giacché l'utilizzo di una singola chiave di recensione per ogni videogioco porterebbe ad una limitazione, quale quella di avere solamente 1 recensione per videogioco.



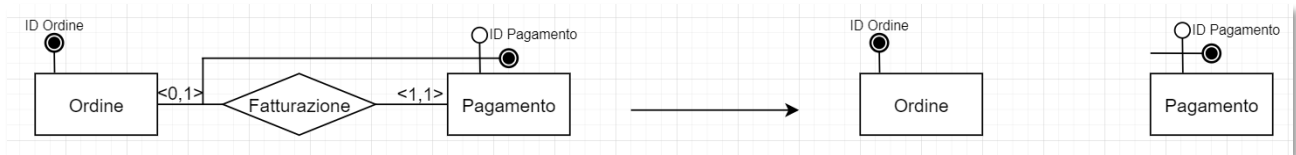
### Magazzino - Ripiano - Videogioco

L'entità Ripiano è identificata da N. Scaffale, N. Ripiano e ID Magazzino proveniente da Magazzino, tramite la relazione "Contiene". La chiave primaria di Magazzino rimane ID Magazzino, mentre la chiave di Ripiano rimane la chiave composta per evitare perdita di dettaglio.



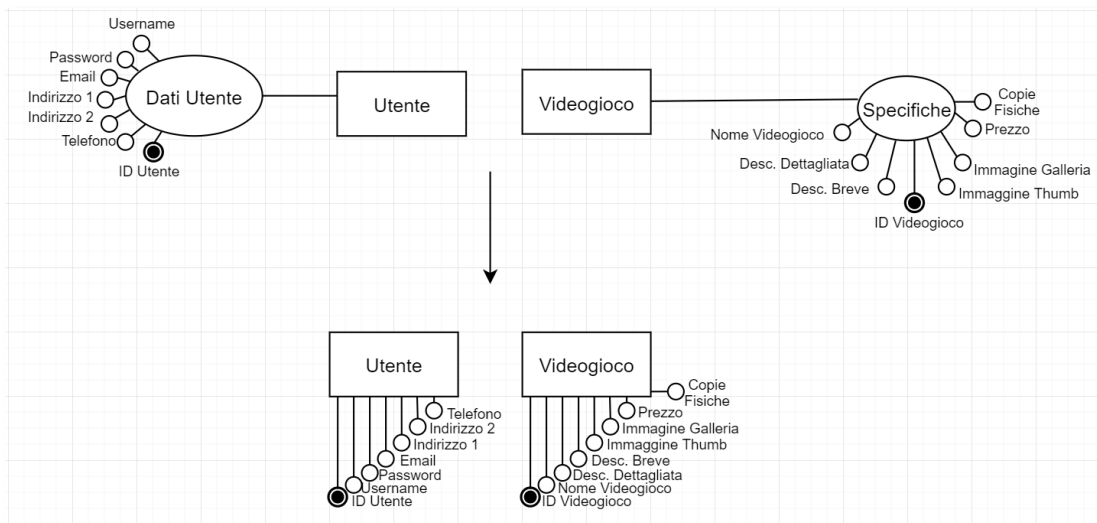
## Ordine - Pagamento

L'entità ordine è identificata da ID\_Ordine e pagamento dalla composizione di ID\_Ordine e ID\_Pagamento: PRIMARY KEY(ID\_Ordine, ID\_Pagamento).



## 3.3 Trasformazione Attributi Composti e Multipli

Nello schema ER abbiamo due attributi multipli che dobbiamo spezzare in sotto-attributi: Dati Utente e Specifiche. Di seguito riportiamo come diventano le due entità a seguito del processo di schematizzazione logica.

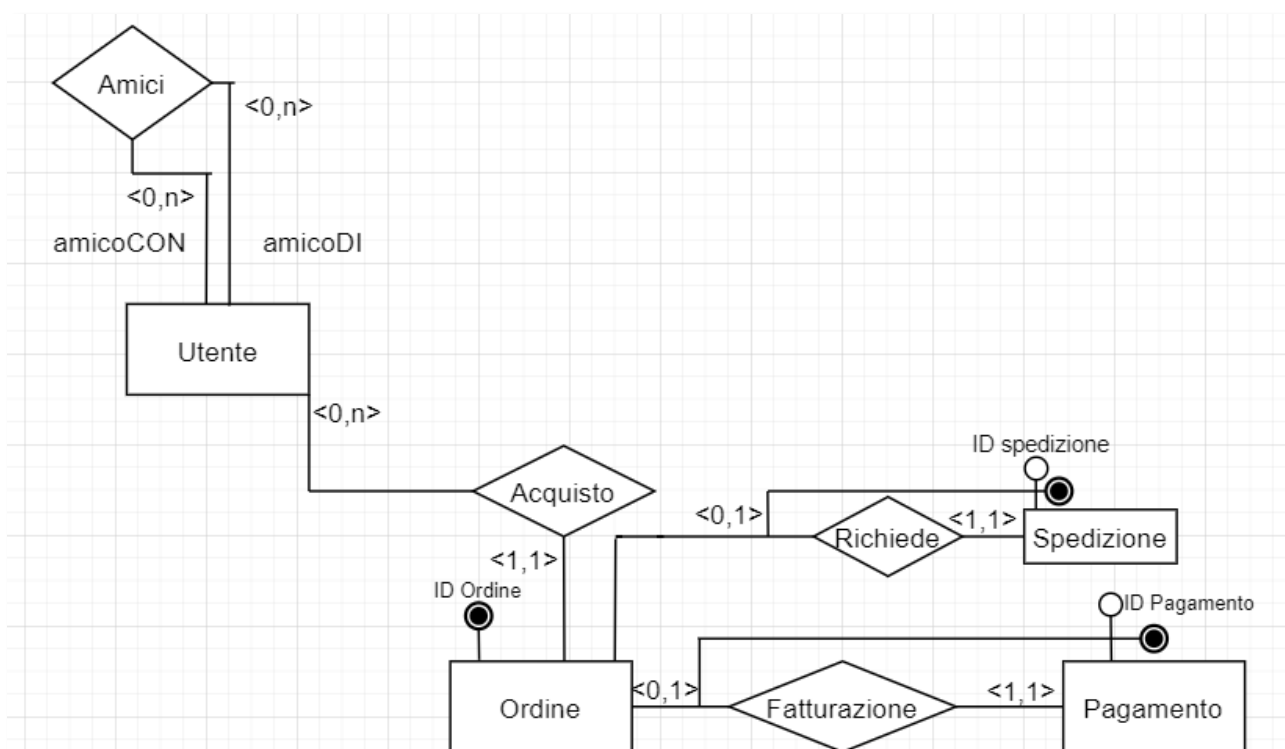


## 3.4 Traduzione di entità e associazioni in schemi relazionali

Per quanto riguarda la traduzione delle entità e delle associazioni in schemi relazionali abbiamo diviso in 3 gruppi lo schema ER:

1. Acquisto
2. Locazione Videogioco
3. Ricerca e Recensione

### 3.4.1 Acquisto



Per quel che riguarda l'utente e il rapporto che questo ha con l'ordine, grazie a "Acquisto" abbiamo utilizzato un tipo di traduzione standard. Nel trattare invece Ordine e tutte le sue associazioni abbiamo deciso di accorpare le associazioni, per altro senza attributi, quali fatturazione e richiede. In questa sezione ci siamo voluti occupare anche della traduzione dell'entità "Amici", decidendo di non accorparla.

**Utente** (IDUtente, Username, Password, Indirizzo1, Indirizzo2, Email, Telefono)

**Amici** (IDUtente, ID\_UTENTE\_AMICO)

FK: IDUtente REFERENCES Utente

AK: Username

AK: Email

AK: Telefono

**Acquisto** (IDUtente, IDOrdine)

FK: IDUtente REFERENCES Utente

FK: IDOrdine REFERENCES Ordine

**Ordine** (IDOrdine)

**Spedizione** (IDSpedizione, IDOrdine)

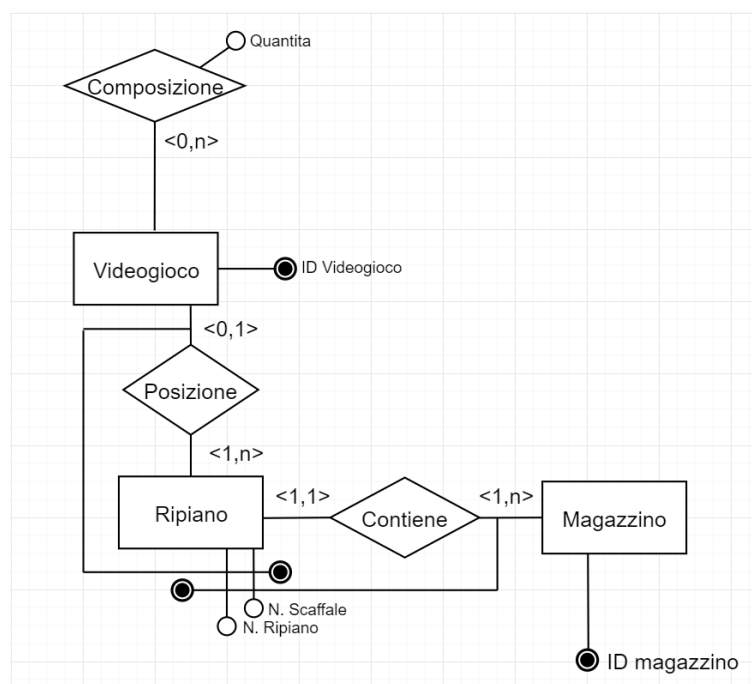
FK: IDOrdine REFERENCES Ordine

**Pagamento** (IDPagamento, IDOrdine, Importo, Data, Numero Carta, Data Scadenza, CCV, Username PayPal, Password PayPal, Selettore Pagamento)

FK: IDOrdine REFERENCES Ordine

### 3.4.2 Posizione Videogioco e Composizione

In questo caso abbiamo accorpato “Contiene” e “Posizione” traducendo tutto il resto.





**Videogioco** (IDVideogioco, Nome Videogioco, Descrizione Breve, Descrizione Dettagliata, Immagine Thumb, Immagini Galleria, Prezzo, Copie Fisiche)

AK: Nome Videogioco

**Ripiano** (N.Scaffale, N.Ripiano, IDVideogioco, IDMagazzino)

FK: IDMagazzino REFERENCES Magazzino

FK: IDVideogioco REFERENCES Videogioco

**Magazzino** (IDMagazzino)

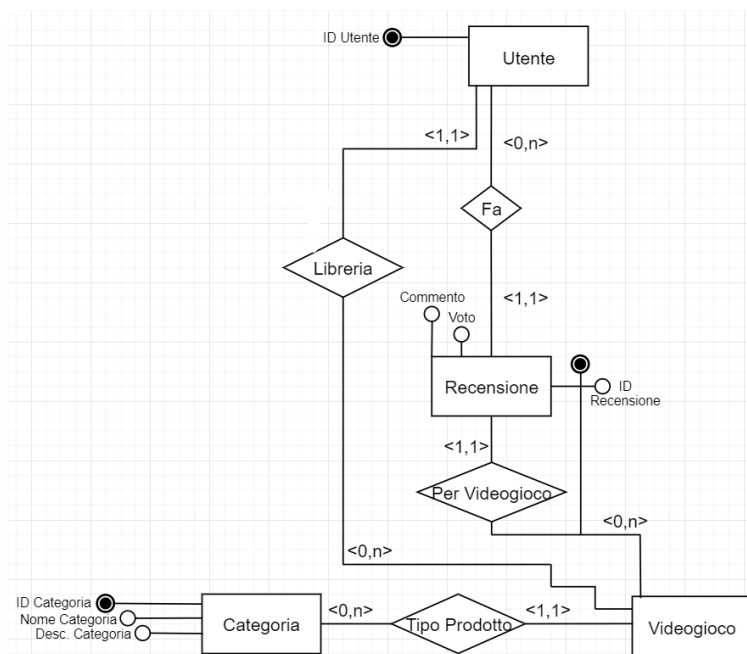
**Composizione** (IDOrdine, IDVideogioco, Quantita)

FK: IDOrdine REFERENCES Ordine

FK: IDVideogioco REFERENCES Videogioco

### 3.4.2 Ricerca e Recensione

Abbiamo accorpato “Per Videogioco” e “Fa” e in seguito tradotto il resto.



**Libreria** (IDUtente, IDVideogioco)

FK: IDUtente REFERENCES Utente

FK: IDVideogioco REFERENCES Videogioco

**Categoria** (IDCategoria, Nome Categoria, Desc. Categoria)

**Recensione** (IDRecensione, Commento, Voto, ID Videogioco, ID\_Utente)

FK: IDVideogioco REFERENCES Videogioco

**Tipo\_Videogioco** (IDCategoria, IDVideogioco)

FK: IDVideogioco REFERENCES Videogioco

FK: IDCategoria REFERENCES Categoria

## 3.4 Normalizzazione

Le relazioni sono in forma normale.

## 4. Dato derivato

Prendiamo ad esempio la procedura di acquisto di un ordine (pagamento) e della visualizzazione del costo complessivo (importo), utilizzando la regola del 20-80.

### Tabella operazioni

Supponiamo di avere ogni giorno una mole di utenti pari al 10.000. Di questi solamente l'1% effettua un acquisto, quindi abbiamo 100 ordini complessivi. Gli ordini in media hanno 1.5 prodotti

Operazioni	Tipo	Frequenza Giornaliera
Inserimento Videogioco in Ordine	Interattiva (I)	150
Visualizzazione costo Ordine (Pagamento)	Interattiva (I)	120

### Carico di lavoro

Gli ordini si generano nel momento in cui un utente inserisce un gioco in un ordine, in caso contrario non esiste alcun ordine per un determinato utente che non ne richiede la creazione. In questi esperimenti riferendoci ad ordine intenderemo sempre gli ordini a cui è legato un pagamento e che quindi sono stati pagati con successo.

Concetto	Tipo	Volume dei dati
Videogioco	Entità (E)	10.000
Ordine	Entità (E)	100
Composizione	Relazione (R)	150

### Tabella degli accessi con dato derivato

Come primo dato abbiamo l'inserimento di un videogioco in un determinato ordine, presupponendo di conoscere già il gioco da aggiungere, così da azzerare il tempo di ricerca.

Come secondo invece abbiamo "Importo" all'interno dell'entità "Pagamento" che è collegato ad Ordine"; importo corrisponde esattamente alla somma dei

costi di tutti i prodotti di un determinato ordine ciascuno moltiplicato per la propria quantità.

Costo Totale	Operazione	Concetto	Accessi	Tipo
5 * 150 = 750	Aggiunta di un Videogioco ad Ordine	Ordine	1	L
		Composizione	1	S
		Videogioco	1	S
3 * 120 = 360	Visualizzazione Importo Pagamento	Ordine	1	L
		Fatturazione	1	L
		Pagamento	1	L

## Tabella degli accessi senza il dato derivato

Costo Totale	Operazione	Concetto	Accessi	Tipo
5 * 150 = 750	Aggiunta di un Videogioco ad Ordine	Ordine	1	L
		Composizione	1	S
		Videogioco	1	S
3 * 120 * 180 = 64800	Visualizzazione Importo Pagamento	Ordine	1	L
		Composizione	1	L
		Videogioco	1	L

**180** → se in un giorno abbiamo 120 venti visualizzazione di importi per gli ordini, in totale significa che si vendono circa 180 giochi al giorno. L'operazione di visualizzazione di importo dell'ordine fatta senza il totale all'interno di pagamento comporta il fatto che per ogni gioco presente nell'ordine bisogna reiterare l'operazione di lettura su Ordine-Composizione-Videogioco. Per fare un esempio: se ho un ordine di 10 giochi, vuol dire che devo fare 10 volte questa terna di letture per sapere l'importo dell'ordine totale, oltre che cercare i suddetti titoli nel db per poi estrapolarne i costi!

Dato lo studio del dato derivato riteniamo quindi fondamentale la presenza di importo totale all'interno del pagamento. Mentre è indifferente che ci sia o meno per l'aggiunta di un determinato titolo in un ordine.

## 5. Progetto Fisico

In questo capitolo ci occuperemo di individuare degli ipotetici indici da accompagnare alle tabelle con più occorrenze e ne analizzeremo il risparmio in termini di accessi.

Di tutto il database sicuramente le tabelle più popolate sono sicuramente utente e videogiochi. Prendiamo come esempio il DB di Steam: 1 miliardo di account registrati, 30 mila videogiochi.

Di un utente quante volte devo vedere i suoi dati? Sicuramente se faccio il login devo andare a cercare la password e l'username/ID\_Utente. E se faccio un ordine? Andrò sicuramente a visualizzare l'indirizzo 1 qualora l'utente chieda la spedizione del prodotto.

E di un Videogioco? Sicuramente ogni volta che cerco ad un nuovo gioco o si apre un banner pubblicitario con all'interno il link per la pagina del videogiochi, il DB viene chiamato dal sito per essere fornito di tutte le informazioni necessarie a generare la pagina del videogiochi in cui stiamo accedendo. È quindi ipotizzabile che sia l'operazione sicuramente più frequente e che necessita quindi assolutamente di essere indicizzata.

Ma quanto risparmieremmo se avessimo un indice? Facciamo due conti:

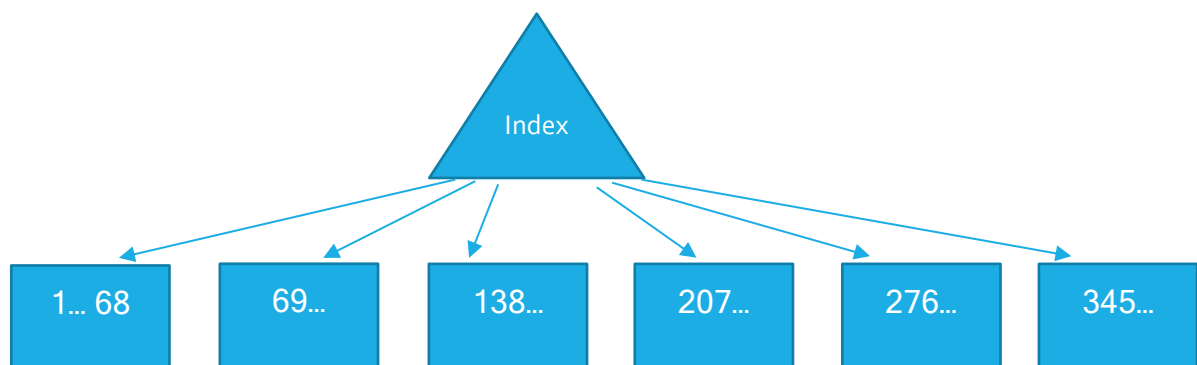
Tabella	Accesso in Lettura	Frequenza
Utente	Accedere ai dati di Login	3.000.000/giorno
	Accedere ai dati di spedizione	10.000/giorno
Videogioco	Accesso a tutti gli attributi per la composizione delle pagine	2.100.000/giorno

Se utilizzassimo blocchi di memoria di 4096 byte per memorizzare unicamente username e password (60 byte per utente), in un blocco potremmo avere un massimo di 68 utenti (68,26). Quanti blocchi avremmo bisogno per contenere 1 miliardo di utenti? 14,7 milioni di blocchi (14.705.882,3529412).

Leggerli in sequenziale costerebbe davvero moltissimo in termini di tempo. Parliamo in media di un Equality Search =  $14,7 \text{ milioni} / 2 * 24.47 \text{ ms} = 179.854.500 \text{ ms} = 50 \text{ ore circa}$ .

Indice o Hash?

A seguito di approfondite ricerche siamo giunti alla conclusione che il metodo più conveniente per organizzare una mole così ampia di dati sia l'utilizzo del Clustered Index B+ Tree. Questo per ottimizzare il più possibile l'equality search su dati come "username", "password" per il login utente e per "ID\_Videogioco" per quel che concerne la ricerca di titoli specifici.



Costo: cerchiamo l'indice, poi andiamo nell'heap file a prendere il dato esatto che stiamo cercando. Il costo per la ricerca dell'indice in questo caso è logaritmico in base F = numero delle foglie moltiplicato per (BR/E) + 1. Aggiungiamo 1 perché contiamo anche la radice dell'albero.

$(\log_{NF} (NB * R / E) + 1) * D$       *E = numero record medio per foglie, R = numero di record per blocchi*

Nel nostro caso di studio un ipotetico costo potrebbe essere:

$F = [(NK * L(k) + NT + * L(p)) / (D * \ln 2)] = \text{circa } 577\,600\,000$

Costo clustered b+ tree =  $1.81803495 * 24,7 \text{ ms} = 44,9 \text{ ms}$

## 6. Tables, Trigger e Dati

```
SET TIME ZONE 'UTC+1';
```

```
CREATE TABLE UTENTE
```

```
(    ID_UTENTE CHAR(5) PRIMARY KEY,  
    USERNAME VARCHAR(30) NOT NULL UNIQUE,  
    PASSWORD VARCHAR(30) NOT NULL,  
    INDIRIZZO1 VARCHAR(60),  
    INDIRIZZO2 VARCHAR(60),  
    EMAIL VARCHAR(60) NOT NULL UNIQUE,  
    TELEFONO VARCHAR(10) NOT NULL UNIQUE,
```

```
    CHECK (LENGTH(USERNAME) >= 6 ),
```

```
    CHECK (LENGTH(PASSWORD) >= 6 )
```

```
);
```

```
CREATE TABLE ORDINE
```

```
(    ID_ORDINE CHAR(5) PRIMARY KEY  
);
```

```
CREATE TABLE ACQUISTO
```

```
(ID_ORDINE CHAR(5),  
  ID_UTENTE CHAR(5),  
  PRIMARY KEY(ID_ORDINE, ID_UTENTE),
```

```
  FOREIGN KEY (ID_ORDINE) REFERENCES ORDINE
```

```
    ON UPDATE CASCADE ON DELETE CASCADE,
```

```
  FOREIGN KEY (ID_UTENTE) REFERENCES UTENTE
```

```
    ON UPDATE CASCADE ON DELETE CASCADE
```

```
);
```

```
CREATE TABLE AMICI
```

```
(    ID_UTENTE CHAR(5) NOT NULL,  
    ID_UTENTE_AMICO CHAR(5) NOT NULL,  
    PRIMARY KEY(ID_UTENTE, ID_UTENTE_AMICO),
```



```
FOREIGN KEY(ID_UTENTE) REFERENCES UTENTE
    ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY(ID_UTENTE_AMICO) REFERENCES UTENTE
    ON UPDATE CASCADE ON DELETE CASCADE,

CHECK(ID_UTENTE != ID_UTENTE_AMICO)

);
```

```
CREATE TABLE PAGAMENTO
(
    ID_PAGAMENTO CHAR(5),
    ID_ORDINE CHAR(5),
    IMPORTO NUMERIC(6,2) NOT NULL,
    DATA DATE NOT NULL,
    NUMERO_CARTA CHAR(16),
    DATA_SCADENZA DATE,
    CCV NUMERIC(3),
    USERNAME_PAYPAL VARCHAR(30),
    PASSWORD_PAYPAL VARCHAR (30),
    SELETTORE_PAGAMENTO VARCHAR(20),
    PRIMARY KEY(ID_PAGAMENTO, ID_ORDINE),

    FOREIGN KEY(ID_ORDINE) REFERENCES ORDINE
    ON UPDATE CASCADE ON DELETE CASCADE,

    CHECK (IMPORTO >= 0)

);
```

```
CREATE TABLE SPEDIZIONE
(
    ID_SPEDIZIONE CHAR(5),
    ID_ORDINE CHAR(5),
    PRIMARY KEY(ID_SPEDIZIONE, ID_ORDINE),

    FOREIGN KEY(ID_ORDINE) REFERENCES ORDINE
    ON UPDATE CASCADE ON DELETE CASCADE

);
```

```
CREATE TABLE CATEGORIA
(
    ID_CATEGORIA CHAR(5) PRIMARY KEY,
    NOME_CATEGORIA VARCHAR (30),
    DESC_CATEGORIA VARCHAR (300)
);
```

```
CREATE TABLE MAGAZZINO
(
    ID_MAGAZZINO CHAR(3) PRIMARY KEY
);
```

```
CREATE TABLE VIDEOGIOCO
(
    ID_VIDEOGIOCO CHAR(5) PRIMARY KEY,
    NOME_VIDEOGIOCO VARCHAR (30) NOT NULL UNIQUE,
    DESCRIZIONE_BREVE VARCHAR(200),
    DESCRIZIONE_DETAGLIATA VARCHAR(1000),
    IMMAGINE_THUMB BOOLEAN,
    IMMAGINE_GALLERIA BOOLEAN,
    PREZZO INTEGER NOT NULL,
    COPIE_FISICHE INTEGER,

    CHECK (COPIE_FISICHE >= 0),
    CHECK (PREZZO >= 0)
);
```

```
CREATE TABLE RIPIANO
(
    ID_VIDEOGIOCO CHAR(5),
    N_RIPIANO CHAR(3),
    N_SCAFFALE CHAR(3),
    ID_MAGAZZINO CHAR(3),
    PRIMARY KEY(ID_MAGAZZINO,N_SCAFFALE,N_RIPIANO, ID_VIDEOGIOCO),

    FOREIGN KEY (ID_MAGAZZINO) REFERENCES MAGAZZINO
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (ID_VIDEOGIOCO) REFERENCES VIDEOGIOCO
        ON UPDATE CASCADE ON DELETE CASCADE
);
```

```

CREATE TABLE COMPOSIZIONE
(
    ID_ORDINE CHAR(5),
    ID_VIDEOGIOCO CHAR(5),
    QUANTITA NUMERIC(2),
    PRIMARY KEY (ID_ORDINE, ID_VIDEOGIOCO),

    FOREIGN KEY (ID_ORDINE) REFERENCES ORDINE
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (ID_VIDEOGIOCO) REFERENCES VIDEOGIOCO
        ON UPDATE CASCADE ON DELETE CASCADE,

    CHECK (QUANTITA > 0)
);

```

```

CREATE TABLE RECENSIONE
(
    ID_RECENSIONE CHAR(5),
    VOTO SMALLINT,
    COMMENTO VARCHAR (500),
    ID_UTENTE CHAR(5),
    ID_VIDEOGIOCO CHAR(5),
    PRIMARY KEY (ID_VIDEOGIOCO, ID_RECENSIONE),

    FOREIGN KEY (ID_UTENTE) REFERENCES UTENTE
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (ID_VIDEOGIOCO) REFERENCES VIDEOGIOCO
        ON UPDATE CASCADE ON DELETE CASCADE,

    CHECK (Voto >= 1 and Voto <= 5)
);

```

```

CREATE TABLE LIBRERIA
(
    ID_UTENTE CHAR(5),
    ID_VIDEOGIOCO CHAR(5),
    PRIMARY KEY (ID_UTENTE, ID_VIDEOGIOCO),
    FOREIGN KEY (ID_UTENTE) REFERENCES UTENTE
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (ID_VIDEOGIOCO) REFERENCES VIDEOGIOCO
        ON UPDATE CASCADE ON DELETE CASCADE
);

```

```

CREATE TABLE TIPO_VIDEOGIOCO(
ID_VIDEOGIOCO CHAR(5),
ID_CATEGORIA CHAR(5),
PRIMARY KEY (ID_VIDEOGIOCO, ID_CATEGORIA),

        FOREIGN KEY (ID_VIDEOGIOCO) REFERENCES VIDEOGIOCO
        ON UPDATE CASCADE ON DELETE CASCADE,
        FOREIGN KEY (ID_CATEGORIA) REFERENCES CATEGORIA
        ON UPDATE CASCADE ON DELETE CASCADE
);

```

## TRIGGER

1. Controllo selettore pagamento: se viene inserito un pagamento con un determinato selettore, il trigger setta come NULL i campi della table pagamento non servono (nel db test è già presente ordine e acquisto relativo al questo pagamento fittizio; per provare il codice basta copiare fino a primo insert).

```

CREATE FUNCTION pagamento() RETURNS trigger AS $pagamento$
BEGIN
    IF (NEW.SELETTORE_PAGAMENTO = 'CARTA DI CREDITO')
        THEN SET NEW.USERNAME_PAYPAL = 'NULL';
        SET NEW.PASSWORD_PAYPAL = 'NULL';
    ELSEIF (NEW.SELETTORE_PAGAMENTO = 'PAYPAL')
        THEN SET NEW.NUMERO_CARTA = 'NULL';
        SET NEW.CCV = 'NULL';
        SET NEW.DATA_SCADENZA = 'NULL';

    END IF;
    RETURN NEW;

END;
$pagamento$ LANGUAGE plpgsql;

CREATE TRIGGER PAGAMENTO
BEFORE INSERT OR UPDATE
ON PAGAMENTO
FOR EACH ROW
EXECUTE PROCEDURE pagamento();

```

```
DELETE FROM PAGAMENTO
WHERE ID_PAGAMENTO = '00001';
```

```
INSERT INTO PAGAMENTO VALUES ('00001','00000',0,'01-01-2001',4119162872354178,'01-12-2027', 228, 'Fluss-lol33@hotmail.it', 'PROVA1', 'CARTA DI CREDITO');
```

```
INSERT INTO ORDINE VALUES ('00000');
INSERT INTO ACQUISTO VALUES ('00000','34003');
INSERT INTO PAGAMENTO VALUES ('00001','00000',0,'01-01-2001',4119162872354178,'01-12-2027', 228, 'Fluss-lol33@hotmail.it', 'PROVA1', 'CARTA DI CREDITO');
```

## Dati da Inserire

```
/*TABLE UTENTE*/
```

```
INSERT INTO UTENTE VALUES ('00001','Marcolino','5BFA6C3DF26D72C5','Via Garibaldi 10',NULL,'marcogamer@libero.it',3894963872);
INSERT INTO UTENTE VALUES ('26951','King_Dark010','E92FCB66EC3F59ED','Via Verdi 67','Piazza Renato 08','France_921@gmail.it',3334963872);
INSERT INTO UTENTE VALUES ('29043','ProGamer97','3FD7EA24A393A66D','Via San Giuseppe 15',NULL,'prog-0123@libero.it',3523945172);
INSERT INTO UTENTE VALUES ('64307','Msmr_dan','872FBF7FFED232C9','Via Rossi 23',NULL,'DanieleVerdi@gmail.it',3614963872);
INSERT INTO UTENTE VALUES ('83442','Stefanoss00','BA2EB42B5759FB89','Via Garibaldi 78',NULL,'steffo-villa@libero.it',3904909322);
INSERT INTO UTENTE VALUES ('02305','Smischeck','CD9FB86B3967169F','Piazza Donatori 33',NULL,'Guglielmo_Glos@yahoo.it',3324963872);
INSERT INTO UTENTE VALUES ('10439','Longo_013','C9B99115BF8F2EFC','Via Garibaldi 67','Viale Coppola 20','MarcoLonghini@gmail.it',3429043872);
INSERT INTO UTENTE VALUES ('18426','Drake_1982','4EAF17A903E4ADA8','Via Rossi 89',NULL,'GibboBom12@hotmail.it',3224003872);
INSERT INTO UTENTE VALUES ('59280','SMF-EI_Ghecko','AEF3CE7C7C9CD5D9','Via Secondino 05',NULL,'MatteGiamma99@hotmail.it',3324963873);
INSERT INTO UTENTE VALUES ('30520','Cavaxey','49D42E2EDB99F347','Via Atlante 66',NULL,'Tom00Baby@libero.it',3324934872);
INSERT INTO UTENTE VALUES ('34003','MaxBubblegum','24FBF2877A88BF77','Via Bonaparte 42',NULL,'Fluss-lol33@hotmail.it',3924094540);
```

```
/*TABLE AMICI*/
```

```
INSERT INTO AMICI VALUES ('00001','26951');
```

```

INSERT INTO AMICI VALUES ('00001','64307');
INSERT INTO AMICI VALUES ('00001','02305');
INSERT INTO AMICI VALUES ('10439','18426');
INSERT INTO AMICI VALUES ('34003','30520');
INSERT INTO AMICI VALUES ('83442','18426');
INSERT INTO AMICI VALUES ('59280','30520');
INSERT INTO AMICI VALUES ('64307','30520');
INSERT INTO AMICI VALUES ('29043','00001');
INSERT INTO AMICI VALUES ('26951','00001');
INSERT INTO AMICI VALUES ('64307','00001');
INSERT INTO AMICI VALUES ('02305','00001');
INSERT INTO AMICI VALUES ('18426','10439');
INSERT INTO AMICI VALUES ('30520','34003');
INSERT INTO AMICI VALUES ('18426','83442');
INSERT INTO AMICI VALUES ('30520','59280');
INSERT INTO AMICI VALUES ('30520','64307');
INSERT INTO AMICI VALUES ('00001','29043');

```

```

/*TABLE ORDINE*/

```

```

INSERT INTO ORDINE VALUES ('00123');
INSERT INTO ORDINE VALUES ('00455');
INSERT INTO ORDINE VALUES ('00489');
INSERT INTO ORDINE VALUES ('00129');
INSERT INTO ORDINE VALUES ('00086');
INSERT INTO ORDINE VALUES ('00732');
INSERT INTO ORDINE VALUES ('00653');
INSERT INTO ORDINE VALUES ('00453');
INSERT INTO ORDINE VALUES ('00321');
INSERT INTO ORDINE VALUES ('00230');
INSERT INTO ORDINE VALUES ('00128');
INSERT INTO ORDINE VALUES ('00085');
INSERT INTO ORDINE VALUES ('00731');
INSERT INTO ORDINE VALUES ('00324');

```

```

/*TABLE ACQUISTO*/

```

```

INSERT INTO ACQUISTO VALUES ('00123','10439');
INSERT INTO ACQUISTO VALUES ('00455','64307');
INSERT INTO ACQUISTO VALUES ('00489','02305');
INSERT INTO ACQUISTO VALUES ('00129','83442');
INSERT INTO ACQUISTO VALUES ('00086','34003');
INSERT INTO ACQUISTO VALUES ('00732','30520');
INSERT INTO ACQUISTO VALUES ('00653','00001');
INSERT INTO ACQUISTO VALUES ('00453','18426');
INSERT INTO ACQUISTO VALUES ('00321','26951');

```

```

INSERT INTO ACQUISTO VALUES ('00230','02305');
INSERT INTO ACQUISTO VALUES ('00128','34003');
INSERT INTO ACQUISTO VALUES ('00085','64307');
INSERT INTO ACQUISTO VALUES ('00731','83442');
INSERT INTO ACQUISTO VALUES ('00324','29043');

```

```

/*TABLE PAGAMENTO*/

```

```

INSERT INTO PAGAMENTO VALUES ('55664','00085',50,'29-01-2019',4119162872354178,'01-12-2027', 228, NULL, NULL, 'CARTA DI CREDITO');
INSERT INTO PAGAMENTO VALUES ('98653','00732',100,'29-09-2019',NULL, NULL, NULL,'Fluss-lol33@hotmail.it','PROVA1', 'PAYPAL');
INSERT INTO PAGAMENTO VALUES ('39287','00123',100,'29-04-2019',5181688183753047,'01-06-2022', 449, NULL, NULL, 'CARTA DI CREDITO');
INSERT INTO PAGAMENTO VALUES ('23847','00455',50,'29-09-2019',4119162872354178,'01-12-2027', 228, NULL, NULL, 'CARTA DI CREDITO');
INSERT INTO PAGAMENTO VALUES ('12093','00489',60,'12-06-2019',5124681429043117,'01-05-2029',736, NULL, NULL, 'CARTA DI CREDITO');
INSERT INTO PAGAMENTO VALUES ('02393','00129',70,'22-08-2019',5510638330917989,'01-12-2026',153, NULL, NULL, 'CARTA DI CREDITO');
INSERT INTO PAGAMENTO VALUES ('30924','00086',90,'01-04-2019',NULL, NULL, NULL,'Fluss-lol33@hotmail.it', 'PROVA2','PAYPAL');
INSERT INTO PAGAMENTO VALUES ('32087','00653',60,'18-09-2019',NULL, NULL, NULL,'MarcoPaypal@gmail.com', 'PAYPAL');
INSERT INTO PAGAMENTO VALUES ('94382','00453',72,'15-03-2018',NULL, NULL, NULL,'GibboBom12@hotmail.it', 'PAYPAL');
INSERT INTO PAGAMENTO VALUES ('02397','00321',80,'18-07-2018',NULL, NULL, NULL,'Luigi&Mario@gmail.com', 'PAYPAL');
INSERT INTO PAGAMENTO VALUES ('48753','00230',50,'20-04-2020',NULL, NULL, NULL,'Guglielmo_Glos@yahoo.it', 'PAYPAL');

```

```

/*TABLE SPEDIZIONE*/

```

```

INSERT INTO SPEDIZIONE VALUES ('83569', '00123');
INSERT INTO SPEDIZIONE VALUES ('95642', '00455');
INSERT INTO SPEDIZIONE VALUES ('92654', '00489');
INSERT INTO SPEDIZIONE VALUES ('58363', '00129');
INSERT INTO SPEDIZIONE VALUES ('04253', '00086');
INSERT INTO SPEDIZIONE VALUES ('92374', '00085');
INSERT INTO SPEDIZIONE VALUES ('82342', '00324');
INSERT INTO SPEDIZIONE VALUES ('34953', '00321');

```

```

/*TABLE CATEGORIA*/

```

```

INSERT INTO CATEGORIA VALUES ('00923','Action RPG','Un action RPG è un videogioco di ruolo a turni che richiede azioni veloci o riflessi pronti da parte del giocatore.');
```



INSERT INTO CATEGORIA VALUES ('00364','Arcade','Un videogioco arcade è un videogioco che si gioca in una postazione pubblica apposta a gettoni o a monete, costituita fisicamente da una macchina posizionata dentro un cabinato.');

INSERT INTO CATEGORIA VALUES ('00430','Avventura','Il videogioco di avventura è un genere di videogioco caratterizzato da esplorazione, risoluzione di enigmi, interazione con personaggi di gioco ed è incentrato sulla narrazione piuttosto che sulle sfide basate sulla prontezza di riflessi.');

INSERT INTO CATEGORIA VALUES ('00023','Battle Royale','Battle royale è una tipologia o modalità dei videogiochi che ha come obiettivo la sopravvivenza in arena player versus player.');

INSERT INTO CATEGORIA VALUES ('00437','Hack and Slash','Hack and slash è un genere di giochi di ruolo da tavolo e di videogiochi basati principalmente sul combattimento con armi a distanza o corpo a corpo.');

INSERT INTO CATEGORIA VALUES ('00579','Videogioco Erotico','Il videogioco erotico è un genere di videogioco caratterizzato dalla presenza di contenuti erotici come immagini osé accompagnate da suoni erotici; tale genere non mostra scene esplicite di sesso o di organi riproduttivi in piena vista, ma è basato su allusioni sessuali.');

INSERT INTO CATEGORIA VALUES ('00279','Gioco di Ruolo','Videogioco di ruolo, spesso abbreviato semplicemente in gioco di ruolo, è un tipo di videogioco che tradizionalmente usa elementi di gioco presi da giochi di ruolo "carta e penna". I videogiochi di ruolo moderni racchiudono una grande varietà di stili di gioco e di motori di gestione.');

INSERT INTO CATEGORIA VALUES ('00192','MMORPG','Un MMORPG è un gioco di ruolo per computer o console che viene svolto online contemporaneamente da più persone reali. Migliaia di giocatori possono interagire interpretando personaggi che si evolvono insieme al mondo persistente che li circonda e in cui vivono.');

INSERT INTO CATEGORIA VALUES ('00743','Roguelike','Roguelike è un termine che indica un particolare tipo di videogiochi di ruolo, caratterizzato da elementi comuni: mappe casuali, morte permanente.');

INSERT INTO CATEGORIA VALUES ('00034','FPS','Lo sparatutto in prima persona è un sottogenere dei videogiochi di tipo sparatutto, che in questo caso adottano una visuale in prima persona. Normalmente, in un videogioco di questo genere, nella parte bassa del campo visivo è possibile vedere la propria arma.');

INSERT INTO CATEGORIA VALUES ('00212','TPS','Spatatutto in terza persona è il termine adottato per indicare quei videogiochi di tipo sparatutto in 3D in cui il personaggio giocante è visibile sullo schermo.');

/\*TABLE VIDEOGIOCO\*/

INSERT INTO VIDEOGIOCO VALUES ('12345','Call of Duty','Serie di videogiochi sparatutto in prima persona','Call of Duty è un FPS in cui è possibile giocare offline in giocatore singolo, oppure online in multigiocatore. Disponibile su PC, smartphone e numerose console.',true,true,70,10);

INSERT INTO VIDEOGIOCO VALUES ('23456','The Binding of Isaac','Videogioco di genere roguelike','Inizialmente il giocatore controlla un bambino di nome Isaac che fugge dalla madre perché le è stato chiesto di sacrificarlo come prova della sua fede. La trama è ispirata alla storia biblica del sacrificio di Isacco.',false,true,30,8);

INSERT INTO VIDEOGIOCO VALUES ('34567','Fortnite','Videogioco di genere Battle Royale','In fortnite ci si trova con altri giocatori in una mappa in cui ci sono armi nascoste da cercare per poter

uccidere ed eliminare gli altri giocatori. Per vincere bisogna rimanere gli unici nella mappa, quindi eliminare tutti gli altri',true,true,10,2);

INSERT INTO VIDEOGIOCO VALUES ('45678','World of Warcraft','World of Warcraft, spesso abbreviato in WoW, è un MMORPG','Come in tutti gli MMORPG è possibile scegliere il proprio personaggio. In questo caso ci sono due principali fazioni in guerra tra loro: alleanza e orda. Se si sceglie di fare parte della prima fazione si può scegliere di essere umani, nani e altre razze "buone"; con la seconda fazione invece su può scegliere di essere orchi, troll o altre razze "malvagie"'.',false,false,50,4);

INSERT INTO VIDEOGIOCO VALUES ('56789','Dark Souls','Dark Souls è una saga di videogiochi action RPG','Dark Souls è un videogioco con visuale in terza persona, quindi è possibile vedere il proprio personaggio. Il giocatore deve esplorare varie ambientazioni in stile dark fantasy e utilizzare armi, magie e strategie per sopravvivere a trappole e nemici.'',true,true,60,15);

INSERT INTO VIDEOGIOCO VALUES ('67890','Darksiders','Saga di videogiochi Hack and Slash','La serie è ambientata su una Terra post-apocalittica, dove gli umani sono quasi estinti e gli angeli combattono contro le orde di demoni per il controllo del mondo. Tra loro ci sono i quattro cavalieri dell'Apocalisse, che hanno il compito di portare equilibrio nel mondo.'',false,true,42,5);

INSERT INTO VIDEOGIOCO VALUES ('78901','XXX Elon Musk','Consigliato','Sconsigliato',true,false,100,1);

INSERT INTO VIDEOGIOCO VALUES ('89012','Borderlands','Serie di videogiochi sparatutto in prima persona','La serie è ambientata in un futuro in cui gli umani sono riusciti a colonizzare altri pianeti. Le colonie sul pianeta Pandora sono state costruite per cercare "La Cripta", luogo mistico che si dice contenga ricchezze inimmaginabili. Il compito del giocatore è andare alla ricerca di questa Cripta.'',true,false,15,0);

INSERT INTO VIDEOGIOCO VALUES ('90123','CS:GO','Counter-Strike: Global Offensive è uno sparatutto in prima persona.','CS:GO è un FPS in cui è possibile giocare solamente online in multiplayer su varie console di gioco.'',false,false,15,3);

INSERT INTO VIDEOGIOCO VALUES ('01234','Battlefield','Serie di videogiochi sparatutto in prima persona','Battlefield è un FPS ambientato tra il XX e il XXII secolo a seconda del titolo. Come gli altri FPS è disponibile su più piattaforme di gioco.'',false,true,40,1);

INSERT INTO VIDEOGIOCO VALUES ('98765','Diablo','Videogioco di genere action RPG','Ambientato in un mondo immaginario medievale, il giocatore si trova a percorrere 16 livelli sotto la città di Tristram, alla fine dei quali ci si trova dentro lo stesso inferno, in cui si affronta Diablo',true,true,20,2);

/\*TABLE MAGAZZINO\*/

INSERT INTO MAGAZZINO VALUES ('054');

INSERT INTO MAGAZZINO VALUES ('004');

INSERT INTO MAGAZZINO VALUES ('012');

INSERT INTO MAGAZZINO VALUES ('075');

INSERT INTO MAGAZZINO VALUES ('042');

/\*TABLE RIPIANO\*/

INSERT INTO RIPIANO VALUES ('78901','014','023','054');

INSERT INTO RIPIANO VALUES ('45678','014','023','004');

```

INSERT INTO RIPIANO VALUES ('98765','032','017','012');
INSERT INTO RIPIANO VALUES ('12345','078','009','075');
INSERT INTO RIPIANO VALUES ('01234','021','011','042');
INSERT INTO RIPIANO VALUES ('56789','004','002','042');
INSERT INTO RIPIANO VALUES ('90123','008','024','054');
INSERT INTO RIPIANO VALUES ('67890','011','025','054');
INSERT INTO RIPIANO VALUES ('34567','012','015','004');
INSERT INTO RIPIANO VALUES ('23456','016','003','004');

```

```

/*TABLE COMPOSIZIONE*/

```

```

INSERT INTO COMPOSIZIONE VALUES ('00123','78901',1);
INSERT INTO COMPOSIZIONE VALUES ('00455','45678',1);
INSERT INTO COMPOSIZIONE VALUES ('00489','98765',1);
INSERT INTO COMPOSIZIONE VALUES ('00129','12345',1);
INSERT INTO COMPOSIZIONE VALUES ('00086','23456',1);
INSERT INTO COMPOSIZIONE VALUES ('00086','56789',1);
INSERT INTO COMPOSIZIONE VALUES ('00732','12345',1);
INSERT INTO COMPOSIZIONE VALUES ('00732','01234',1);
INSERT INTO COMPOSIZIONE VALUES ('00653','56789',1);
INSERT INTO COMPOSIZIONE VALUES ('00453','12345',1);
INSERT INTO COMPOSIZIONE VALUES ('00453','89012',1);
INSERT INTO COMPOSIZIONE VALUES ('00321','56789',1);
INSERT INTO COMPOSIZIONE VALUES ('00321','90123',1);
INSERT INTO COMPOSIZIONE VALUES ('00230','45678',1);
INSERT INTO COMPOSIZIONE VALUES ('00128','90123',1);
INSERT INTO COMPOSIZIONE VALUES ('00128','01234',1);
INSERT INTO COMPOSIZIONE VALUES ('00128','98765',1);
INSERT INTO COMPOSIZIONE VALUES ('00085','12345',1);
INSERT INTO COMPOSIZIONE VALUES ('00085','34567',1);
INSERT INTO COMPOSIZIONE VALUES ('00731','67890',1);
INSERT INTO COMPOSIZIONE VALUES ('00324','45678',1);

```

```

/*TABLE RECENSIONE*/

```

```

INSERT INTO RECENSIONE VALUES ('10001',5,NULL,'34003','56789');
INSERT INTO RECENSIONE VALUES ('10002',4,NULL,'34003','90123');
INSERT INTO RECENSIONE VALUES ('10003',5,'Bello.','30520','12345');
INSERT INTO RECENSIONE VALUES ('10004',1,NULL,'83442','67890');
INSERT INTO RECENSIONE VALUES ('10005',3,NULL,'10439','78901');
INSERT INTO RECENSIONE VALUES ('10006',3,NULL,'10439','98765');
INSERT INTO RECENSIONE VALUES ('10007',4,NULL,'30520','01234');
INSERT INTO RECENSIONE VALUES ('10008',3,NULL,'30520','45678');

```

```

/*TABLE LIBRERIA*/

```

```

INSERT INTO LIBRERIA VALUES ('10439','78901');

```

```

INSERT INTO LIBRERIA VALUES ('64307','45678');
INSERT INTO LIBRERIA VALUES ('02305','98765');
INSERT INTO LIBRERIA VALUES ('83442','12345');
INSERT INTO LIBRERIA VALUES ('34003','23456');
INSERT INTO LIBRERIA VALUES ('34003','56789');
INSERT INTO LIBRERIA VALUES ('30520','12345');
INSERT INTO LIBRERIA VALUES ('30520','01234');
INSERT INTO LIBRERIA VALUES ('00001','56789');
INSERT INTO LIBRERIA VALUES ('18426','12345');
INSERT INTO LIBRERIA VALUES ('18426','89012');
INSERT INTO LIBRERIA VALUES ('26951','56789');
INSERT INTO LIBRERIA VALUES ('26951','90123');
INSERT INTO LIBRERIA VALUES ('02305','45678');
INSERT INTO LIBRERIA VALUES ('64307','12345');
INSERT INTO LIBRERIA VALUES ('64307','34567');
INSERT INTO LIBRERIA VALUES ('83442','67890');
INSERT INTO LIBRERIA VALUES ('29043','45678');

```

/\*TIPO PRODOTTO\*/

```

INSERT INTO TIPO_VIDEOGIOCO VALUES('12345','00034');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('23456','00743');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('34567','00023');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('45678','00192');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('56789','00923');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('67890','00437');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('78901','00579');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('89012','00212');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('90123','00034');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('01234','00034');
INSERT INTO TIPO_VIDEOGIOCO VALUES ('98765','00923');

```

## 7. Interrogazioni DBMS

1. Trovare gli username degli amici dell'utente con username "Marcolino"

```
SELECT USERNAME FROM UTENTE
WHERE ID_UTENTE IN (
    SELECT ID_UTENTE_AMICO FROM AMICI
    WHERE ID_UTENTE = (
        SELECT ID_UTENTE FROM UTENTE
        WHERE USERNAME = 'Marcolino'))
```



	username
	character varying (30)
1	King_Dark010
2	ProGamer97
3	Msmr_dan
4	Smischeck

2. Trovare tutti i videogiochi acquistati dall'utente con username "MaxBubblegum"

```
SELECT DISTINCT V.NOME_VIDEOGIOCO
FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP, ORDINE AS O, UTENTE AS U, ACQUISTO AS ACQ, PAGAMENTO AS PAG
WHERE U.USERNAME = 'MaxBubblegum'
AND ACQ.ID_UTENTE = U.ID_UTENTE
AND ACQ.ID_ORDINE = O.ID_ORDINE
AND O.ID_ORDINE = PAG.ID_ORDINE
AND COMP.ID_ORDINE = O.ID_ORDINE
AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
```



	nome_videogioco
	character varying (30)
1	Dark Souls
2	The Binding of Isaac

3. Trovare la spesa totale dell'utente con username "MaxBubblegum"

```
SELECT U.USERNAME, SUM(V.PREZZO) AS IMPORTO
FROM ORDINE AS O, UTENTE AS U, COMPOSIZIONE AS COMP, VIDEOGIOCO AS V, ACQUISTO AS ACQ, PAGAMENTO AS PAG
```

```

WHERE U.USERNAME = 'MaxBubblegum'
AND ACQ.ID_UTENTE = U.ID_UTENTE
AND ACQ.ID_ORDINE = O.ID_ORDINE
AND O.ID_ORDINE = PAG.ID_ORDINE
AND COMP.ID_ORDINE = O.ID_ORDINE
AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
GROUP BY U.USERNAME

```

	username character varying (30)	importo bigint
1	MaxBubblegum	90

**4. Trovare tutti gli utenti che hanno acquistato un videogioco della categoria "FPS"**

```

SELECT U.USERNAME
FROM UTENTE AS U, ORDINE AS O, PAGAMENTO AS PAG,
VIDEOGIOCO AS V, TIPO_VIDEOGIOCO AS TIPO, CATEGORIA AS CAT,
COMPOSIZIONE AS COMP, ACQUISTO AS ACQ
WHERE U.ID_UTENTE = ACQ.ID_UTENTE
AND ACQ.ID_ORDINE = O.ID_ORDINE
AND O.ID_ORDINE = PAG.ID_ORDINE
AND COMP.ID_ORDINE = O.ID_ORDINE
AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
AND V.ID_VIDEOGIOCO = TIPO.ID_VIDEOGIOCO
AND TIPO.ID_CATEGORIA = CAT.ID_CATEGORIA
AND CAT.NOME_CATEGORIA = 'FPS'
GROUP BY U.USERNAME

```

	username character varying (30)
1	Cavaxey
2	Drake_1982
3	King_Dark010
4	Msmr_dan
5	Stefanoss00

**5. Quantità di videogiochi venduti di ciascun videogioco col relativo nome**

```
SELECT      V.NOME_VIDEOGIOCO      AS      VIDEOGIOCO,
SUM(COMP.QUANTITA) AS COPIE_VENDUTE
FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP, ORDINE AS
ORD, PAGAMENTO AS PAG
WHERE V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
AND COMP.ID_ORDINE = ORD.ID_ORDINE
AND PAG.ID_ORDINE = ORD.ID_ORDINE
GROUP BY V.ID_VIDEOGIOCO
```

	videogioco character varying (30)	copie_vendute numeric
1	CS:GO	1
2	Borderlands	1
3	World of Warcraft	2
4	Battlefield	1
5	Call of Duty	4
6	Dark Souls	3
7	Diablo	1
8	The Binding of Isaac	1
9	XXX Elon Musk	1
10	Fortnite	1

**6. Selezionare tutti i giochi che sono stati spediti con relativo nome utente e data di chi li ha acquistati**

```
SELECT V.NOME_VIDEOGIOCO AS VIDEOGIOCO, PAG.DATA AS
DATA_SPEDIZIONE, U.USERNAME AS UTENTE
FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP, ORDINE AS
ORD, PAGAMENTO AS PAG, SPEDIZIONE AS SHIP, UTENTE AS U,
ACQUISTO AS ACQ
WHERE U.ID_UTENTE = ACQ.ID_UTENTE
AND ACQ.ID_ORDINE = ORD.ID_ORDINE
AND PAG.ID_ORDINE = ORD.ID_ORDINE
AND SHIP.ID_ORDINE = ORD.ID_ORDINE
AND COMP.ID_ORDINE = ORD.ID_ORDINE
AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
GROUP BY V.ID_VIDEOGIOCO, PAG.DATA, U.USERNAME
```

	videogioco character varying (30)	data_spedizione date	utente character varying (30)
1	Call of Duty	2019-01-29	Msmr_dan
2	Call of Duty	2019-08-22	Stefanoss00
3	The Binding of Isaac	2019-04-01	MaxBubblegum
4	Fortnite	2019-01-29	Msmr_dan
5	World of Warcraft	2019-09-29	Msmr_dan
6	Dark Souls	2018-07-18	King_Dark010
7	Dark Souls	2019-04-01	MaxBubblegum
8	XXX Elon Musk	2019-04-29	Longo_013
9	CS:GO	2018-07-18	King_Dark010
10	Diablo	2019-06-12	Smischeck

## 7. Posizione di ogni gioco nel magazzino

```

SELECT  V.NOME_VIDEOGIOCO  AS  NOME_VIDEOGIOCO,
V.COPIE_FISICHE  AS  COPIE_VIDEOGIOCO,  R.N_RIPIANO  AS
RIPIANO, R.N_SCAFFALE  AS  SCAFFALE, MAG.ID_MAGAZZINO  AS
MAGAZZINO
FROM VIDEOGIOCO  AS  V, RIPIANO  AS  R, MAGAZZINO  AS  MAG
WHERE V.ID_VIDEOGIOCO  =  R.ID_VIDEOGIOCO
AND R.ID_MAGAZZINO  =  MAG.ID_MAGAZZINO

```

	nome_videogioco character varying (30)	copie_videogioco integer	ripiano character (3)	scaffale character (3)	magazzino character (3)
1	XXX Elon Musk		1 014	023	054
2	World of Warcraft		4 014	023	004
3	Diablo		2 032	017	012
4	Call of Duty		10 078	009	075
5	Battlefield		1 021	011	042
6	Dark Souls		15 004	002	042
7	CS:GO		3 008	024	054
8	Darksiders		5 011	025	054
9	Fortnite		2 012	015	004
10	The Binding of Isaac		8 016	003	004

## 8. Tutti i giochi di Maxbubblegum in libreria

```

SELECT U.USERNAME  AS  GIOCATORI, V.NOME_VIDEOGIOCO  AS
TITOLO_IN_LIBRERIA
FROM VIDEOGIOCO  AS  V, LIBRERIA  AS  LIB, UTENTE  AS  U
WHERE U.USERNAME  =  'MaxBubblegum'
AND LIB.ID_UTENTE  =  U.ID_UTENTE
AND LIB.ID_VIDEOGIOCO  =  V.ID_VIDEOGIOCO

```



	giocatori character varying (30)	titolo_in_libreria character varying (30)
1	MaxBubblegum	The Binding of Isaac
2	MaxBubblegum	Dark Souls

## 9. Giochi con voto >= 4

```
SELECT V.NOME_VIDEOGIOCO, R.VOTO
FROM VIDEOGIOCO AS V, RECENSIONE AS R
WHERE V.ID_VIDEOGIOCO = R.ID_VIDEOGIOCO
AND R.VOTO >= 4
```

	nome_videogioco character varying (30)	voto smallint
1	Dark Souls	5
2	CS:GO	4
3	Call of Duty	5

## 10. Giochi in comune fra più utenti

```
SELECT DISTINCT V.NOME_VIDEOGIOCO, U1.USERNAME AS
UTENTE_1, U2.USERNAME AS UTENTE_2
FROM VIDEOGIOCO AS V, UTENTE AS U1, UTENTE AS U2,
LIBRERIA AS C1, LIBRERIA AS C2
WHERE U1.ID_UTENTE = C1.ID_UTENTE
AND U2.ID_UTENTE = C2.ID_UTENTE
AND V.ID_VIDEOGIOCO = C1.ID_VIDEOGIOCO
AND V.ID_VIDEOGIOCO = C2.ID_VIDEOGIOCO
AND U1.ID_UTENTE != U2.ID_UTENTE
GROUP BY V.NOME_VIDEOGIOCO, U1.USERNAME, U2.USERNAME
```

	nome_videogioco character varying (30)	utente_1 character varying (30)	utente_2 character varying (30)
1	Call of Duty	Cavaxey	Drake_1982
2	Call of Duty	Cavaxey	Msmr_dan
3	Call of Duty	Cavaxey	Stefanoss00
4	Call of Duty	Drake_1982	Cavaxey
5	Call of Duty	Drake_1982	Msmr_dan
6	Call of Duty	Drake_1982	Stefanoss00
7	Call of Duty	Msmr_dan	Cavaxey
8	Call of Duty	Msmr_dan	Drake_1982
9	Call of Duty	Msmr_dan	Stefanoss00
10	Call of Duty	Stefanoss00	Cavaxey
11	Call of Duty	Stefanoss00	Drake_1982
12	Call of Duty	Stefanoss00	Msmr_dan
13	Dark Souls	King_Dark010	Marcolino
14	Dark Souls	King_Dark010	MaxBubblegum
15	Dark Souls	Marcolino	King_Dark010
16	Dark Souls	Marcolino	MaxBubblegum
17	Dark Souls	MaxBubblegum	King_Dark010
18	Dark Souls	MaxBubblegum	Marcolino
19	World of Warcraft	Msmr_dan	ProGamer97
20	World of Warcraft	Msmr_dan	Smischeck
21	World of Warcraft	ProGamer97	Msmr_dan
22	World of Warcraft	ProGamer97	Smischeck
23	World of Warcraft	Smischeck	Msmr_dan
24	World of Warcraft	Smischeck	ProGamer97

# 11. Quante volte è stato acquistato "Dark Souls"

```

SELECT COUNT (V.NOME_VIDEOGIOCO)
FROM VIDEOGIOCO AS V, ORDINE AS ORD, COMPOSIZIONE AS
COMP, PAGAMENTO AS PAG
WHERE V.NOME_VIDEOGIOCO = 'Dark Souls'
AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO
AND COMP.ID_ORDINE = ORD.ID_ORDINE
AND ORD.ID_ORDINE = PAG.ID_ORDINE

```

	count bigint
1	3

## 12. Categorie di videogiochi e videogiochi associati

```
SELECT CAT.NOME_CATEGORIA, VID.NOME_VIDEOGIOCO
FROM CATEGORIA AS CAT, TIPO_VIDEOGIOCO AS TIP,
VIDEOGIOCO AS VID
WHERE VID.ID_VIDEOGIOCO = TIP.ID_VIDEOGIOCO
AND TIP.ID_CATEGORIA = CAT.ID_CATEGORIA
```

	nome_categoria character varying (30)	nome_videogioco character varying (30)
1	FPS	Call of Duty
2	Roguelike	The Binding of Isaac
3	Battle Royale	Fortnite
4	MMORPG	World of Warcraft
5	Action RPG	Dark Souls
6	Hack and Slash	Darksiders
7	Videogioco Erotico	XXX Elon Musk
8	TPS	Borderlands
9	FPS	CS:GO
10	FPS	Battlefield
11	Action RPG	Diablo

## DELETE

### 1. Eliminazione di un account

```
KILLATO MAX BUBBLEGUM
DELETE FROM UTENTE
WHERE ID_UTENTE = '34003'
```

### 2. Eliminazione di un videogioco

```
DELETE FROM VIDEOGIOCO
WHERE NOME_VIDEOGIOCO = 'Diablo'
```

## UPDATE

### 1. Aggiornamento valori relativi ad un videogioco

```
UPADTE VIDEOGIOCO
SET COPIE_FISICHE = 30, PREZZO = PREZZO + 5
WHERE NOME_VIDEOGIOCO = 'COD'
```

## 8. JDBC

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Statement;

public class main {
    public static void main(String[] args) throws IOException {
        try {
            BufferedReader Input = new BufferedReader(new
InputStreamReader(System.in));

            String operation1 = "1. Trovare gli username degli amici
dell'utente con username = Marcolino";
            String operation2 = "2. Trovare tutti i videogiochi
acquistati dall'utente con username = MaxBubblegum";
            String operation3 = "3. Calcolo spesa totale dei giochi
acquistati da MaxBubblegum";
            String operation4 = "4. Individuare tutti gli utenti che
hanno acquistato almeno 1 gioco di categoria FPS";
            String operation5 = "5. Quantità di videogiochi venduti
di ciascun videogioco col relativo nome";
            String operation6 = "6. Selezionare tutti i giochi che
sono stati spediti con relativo nome utente e data di chi li ha
acquistati";
            String operation7 = "7. Posizione di ogni gioco nel
magazzino ";
            String operation8 = "8. Tutti i giochi di max in
libreria ";
            String operation9 = "9. Giochi con voto >= 4";
            String operation10 = "10. Giochi in comune fra più
utenti ";
            String operation11 = "11. Quante volte è stato
acquistato 'Dark Souls'";
            String operation12 = "12. Categorie di videogiochi e
videogiochi associati";
            String operation13 = "13. Digita 'esci' per uscire";

            String sql1 = "SELECT USERNAME FROM UTENTE\r\n" + "WHERE
ID_UTENTE IN (\r\n"
                        + "          SELECT ID_UTENTE_AMICO FROM
AMICI\r\n" + "          WHERE ID_UTENTE = (\r\n"
```

```

        + "                SELECT ID_UTENTE FROM
UTENTE\r\n" + "                WHERE USERNAME = 'Marcolino'))\r\n"
        + """;

String sql2 = "SELECT DISTINCT V.NOME_VIDEOGIOCO\r\n"
        + "FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP,
ORDINE AS O, UTENTE AS U, ACQUISTO AS ACQ, PAGAMENTO AS PAG\r\n"
        + "WHERE U.USERNAME = 'MaxBubblegum'\r\n" +
"AND ACQ.ID_UTENTE = U.ID_UTENTE\r\n"
        + "AND ACQ.ID_ORDINE = O.ID_ORDINE\r\n" + "AND
O.ID_ORDINE = PAG.ID_ORDINE\r\n"
        + "AND COMP.ID_ORDINE = O.ID_ORDINE\r\n" +
"AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO\r\n" + """;

String sql3 = "SELECT U.USERNAME, SUM(V.PREZZO) AS
IMPORTO\r\n"
        + "FROM ORDINE AS O, UTENTE AS U, COMPOSIZIONE
AS COMP, VIDEOGIOCO AS V, ACQUISTO AS ACQ, PAGAMENTO AS PAG\r\n"
        + "WHERE U.USERNAME = 'MaxBubblegum'\r\n" +
"AND ACQ.ID_UTENTE = U.ID_UTENTE\r\n"
        + "AND ACQ.ID_ORDINE = O.ID_ORDINE\r\n" + "AND
O.ID_ORDINE = PAG.ID_ORDINE\r\n"
        + "AND COMP.ID_ORDINE = O.ID_ORDINE\r\n" +
"AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO\r\n"
        + "GROUP BY U.USERNAME\r\n" + """;

String sql4 = "SELECT U.USERNAME\r\n"
        + "FROM UTENTE AS U, ORDINE AS O, PAGAMENTO AS
PAG, VIDEOGIOCO AS V, TIPO_VIDEOGIOCO AS TIPO, CATEGORIA AS CAT,
COMPOSIZIONE AS COMP, ACQUISTO AS ACQ\r\n"
        + "WHERE U.ID_UTENTE = ACQ.ID_UTENTE\r\n" +
"AND ACQ.ID_ORDINE = O.ID_ORDINE\r\n"
        + "AND O.ID_ORDINE = PAG.ID_ORDINE\r\n" + "AND
COMP.ID_ORDINE = O.ID_ORDINE\r\n"
        + "AND V.ID_VIDEOGIOCO =
COMP.ID_VIDEOGIOCO\r\n" + "AND V.ID_VIDEOGIOCO = TIPO.ID_VIDEOGIOCO\r\n"
        + "AND TIPO.ID_CATEGORIA =
CAT.ID_CATEGORIA\r\n" + "AND CAT.NOME_CATEGORIA = 'FPS'\r\n"
        + "GROUP BY U.USERNAME\r\n" + """;

String sql5 = "SELECT V.NOME_VIDEOGIOCO AS VIDEOGIOCO,
SUM(COMP.QUANTITA) AS COPIE_VENDUTE\r\n"
        + "FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP,
ORDINE AS ORD, PAGAMENTO AS PAG\r\n"
        + "WHERE V.ID_VIDEOGIOCO =
COMP.ID_VIDEOGIOCO\r\n" + "AND COMP.ID_ORDINE = ORD.ID_ORDINE\r\n"
        + "AND PAG.ID_ORDINE = ORD.ID_ORDINE \r\n" +
"GROUP BY V.ID_VIDEOGIOCO\r\n" + """;

```

```

String sql6 = "SELECT V.NOME_VIDEOGIOCO AS VIDEOGIOCO,
PAG.DATA AS DATA_SPEDIZIONE, U.USERNAME AS UTENTE \r\n"
+ "FROM VIDEOGIOCO AS V, COMPOSIZIONE AS COMP,
ORDINE AS ORD, PAGAMENTO AS PAG, SPEDIZIONE AS SHIP, UTENTE AS U,
ACQUISTO AS ACQ\r\n"
+ "WHERE U.ID_UTENTE = ACQ.ID_UTENTE\r\n" +
"AND ACQ.ID_ORDINE = ORD.ID_ORDINE\r\n"
+ "AND PAG.ID_ORDINE = ORD.ID_ORDINE\r\n" +
"AND SHIP.ID_ORDINE = ORD.ID_ORDINE\r\n"
+ "AND COMP.ID_ORDINE = ORD.ID_ORDINE\r\n" +
"AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO \r\n"
+ "GROUP BY V.ID_VIDEOGIOCO, PAG.DATA,
U.USERNAME\r\n" + "";
String sql7 = "SELECT V.NOME_VIDEOGIOCO AS
NOME_VIDEOGIOCO, V.COPIE_FISCHE AS COPIE_VIDEOGIOCO, R.N_RIPIANO AS
RIPIANO, R.N_SCAFFALE AS SCAFFALE, MAG.ID_MAGAZZINO AS MAGAZZINO\r\n"
+ "FROM VIDEOGIOCO AS V, RIPIANO AS R,
MAGAZZINO AS MAG \r\n"
+ "WHERE V.ID_VIDEOGIOCO =
R.ID_VIDEOGIOCO\r\n" + "AND R.ID_MAGAZZINO = MAG.ID_MAGAZZINO\r\n" + "";
String sql8 = "SELECT U.USERNAME AS GIOCATORI,
V.NOME_VIDEOGIOCO AS TITOLO_IN_LIBRERIA\r\n"
+ "FROM VIDEOGIOCO AS V, LIBRERIA AS LIB,
UTENTE AS U\r\n" + "WHERE U.USERNAME = 'MaxBubblegum'\r\n"
+ "AND LIB.ID_UTENTE = U.ID_UTENTE\r\n" + "AND
LIB.ID_VIDEOGIOCO = V.ID_VIDEOGIOCO\r\n" + "";
String sql9 = "SELECT V.NOME_VIDEOGIOCO, R.VOTO\r\n" +
"FROM VIDEOGIOCO AS V, RECENSIONE AS R\r\n"
+ "WHERE V.ID_VIDEOGIOCO =
R.ID_VIDEOGIOCO\r\n" + "AND R.VOTO >= 4\r\n" + "";

String sql10 = "SELECT DISTINCT V.NOME_VIDEOGIOCO,
U1.USERNAME AS UTENTE_1, U2.USERNAME AS UTENTE_2\r\n"
+ "FROM VIDEOGIOCO AS V, UTENTE AS U1, UTENTE
AS U2, LIBRERIA AS C1, LIBRERIA AS C2\r\n"
+ "WHERE U1.ID_UTENTE = C1.ID_UTENTE\r\n" +
"AND U2.ID_UTENTE = C2.ID_UTENTE\r\n"
+ "AND V.ID_VIDEOGIOCO = C1.ID_VIDEOGIOCO\r\n"
+ "AND V.ID_VIDEOGIOCO = C2.ID_VIDEOGIOCO\r\n"
+ "AND U1.ID_UTENTE != U2.ID_UTENTE\r\n"
+ "GROUP BY V.NOME_VIDEOGIOCO, U1.USERNAME,
U2.USERNAME\r\n" + "";
String sql11 = "SELECT COUNT (V.NOME_VIDEOGIOCO)\r\n"
+ "FROM VIDEOGIOCO AS V, ORDINE AS ORD,
COMPOSIZIONE AS COMP, PAGAMENTO AS PAG\r\n"
+ "WHERE V.NOME_VIDEOGIOCO = 'Dark Souls'\r\n"
+ "AND V.ID_VIDEOGIOCO = COMP.ID_VIDEOGIOCO\r\n"
+ "AND COMP.ID_ORDINE = ORD.ID_ORDINE\r\n" +
"AND ORD.ID_ORDINE = PAG.ID_ORDINE";

```

```

        String sql12 = "SELECT CAT.NOME_CATEGORIA,
VID.NOME_VIDEOGIOCO\r\n"
            + "FROM CATEGORIA AS CAT, TIPO_VIDEOGIOCO AS
TIP, VIDEOGIOCO AS VID\r\n"
            + "WHERE VID.ID_VIDEOGIOCO =
TIP.ID_VIDEOGIOCO\r\n" + "AND TIP.ID_CATEGORIA = CAT.ID_CATEGORIA\r\n";

        String scelta = "0";

        boolean menu = true;
        while (menu) {
            System.out.println("Scegliere Query: " + "\n" +
operation1 + "\n" + operation2 + "\n" + operation3
            + "\n" + operation4 + "\n" + operation5
+ "\n" + operation6 + "\n" + operation7 + "\n"
            + operation8 + "\n" + operation9 + "\n"
+ operation10 + "\n" + operation11 + "\n" + operation12
            + "\n" + operation13);
            scelta = Input.readLine();

            switch (scelta) {

                case "1":
                    talkingterminal(sql1);
                    break;
                case "2":
                    talkingterminal(sql2);
                    break;

                case "3":
                    talkingterminal(sql3);
                    break;

                case "4":
                    talkingterminal(sql4);
                    break;

                case "5":
                    talkingterminal(sql5);
                    break;

                case "6":
                    talkingterminal(sql6);
                    break;

                case "7":
                    talkingterminal(sql7);
                    break;

                case "8":

```

```

        talkingterminal(sql8);
        break;

    case "9":
        talkingterminal(sql9);
        break;

    case "10":
        talkingterminal(sql10);
        break;

    case "11":
        talkingterminal(sql11);
        break;

    case "12":
        talkingterminal(sql12);
        break;

    case "esci":
        String sql13 = "esci";
        talkingterminal(sql13);
        menu = false;
        break;
    default:
        System.out.println("Scelta della query non
disponibile");
    }

}

} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
}
}

public static void talkingterminal(String sql) throws SQLException,
ClassNotFoundException {
    Class.forName("org.postgresql.Driver"); // Load the Driver
    Connection conn = DriverManager

        .getConnection("jdbc:postgresql://localhost:5432/Digital Delivery &
Shipping N.32", "postgres", "1999");

    if (sql == "esci") {
        conn.close();
    }
}

```



```

PreparedStatement stmt = conn.prepareStatement(sql);
ResultSet rs = stmt.executeQuery();
ResultSetMetaData rsmd = rs.getMetaData();
int columnsNumber = rsmd.getColumnCount();

String header = "0";
String datacontent = "";
int cont = 0;

for (int i = 1; i <= columnsNumber; i++) {
    header = header + "\t\t" + rsmd.getColumnName(i);
}
System.out.println(header);

System.out.println("_____
");

while (rs.next()) {
    cont++;
    for (int i = 1; i <= columnsNumber; i++) {
        datacontent = datacontent + "\t\t" +
rs.getString(i);
    }

    System.out.print(cont + datacontent + "\n");
    datacontent = "";
}

System.out.println("_____
");
}
}

```