Il Leone Rampante: Gioco di Ruolo dal Vivo nel Veneto

Abstract

Con LARP (“Live Action Role Play”, in italiano Gioco di Ruolo dal vivo, GRV) si intende un gioco interattivo dove i giocatori impersonano fisicamente dei personaggi inventati all’interno di un contesto narrativo. L’immedesimazione è accentuata tramite l’uso di costumi e scenografie a tema, e le azioni e reazioni di ogni giocatore vengono rappresentate fisicamente in base alle caratteristiche del relativo personaggio associato.
L’associazione “Il Leone Rampante”, tramite una web app collegata ad un database, gestisce gli eventi di GRV da loro svolti nel Veneto. I dettagli di ogni evento andranno conservati, sia per motivi intellettuali che per motivi tecnici elencati successivamente.
Gli utenti potranno visualizzare i personaggi da loro interpretati, i LARP a cui hanno partecipato, e prenotarsi per quelli futuri. Ogni evento ha associato una lista di personaggi, ognuno con specifiche caratteristiche, il quale l’utente dovrà scegliere alla prenotazione. Determinati eventi, inoltre, saranno svolti nell’arco di più giorni, e sarà quindi necessario gestire anche eventuali pernottamenti e allergeni per pranzi/cene.

Analisi dei requisiti

Descrizione Testuale

La base di dati vuole tenere in considerazione di LARP sequel, prequel di altri e di riedizioni. La partecipazione a LARP sequel o prequel non avrà comunque vincoli particolari. Si vuole evidenziare la differenza tra LARP ed evento tramite un esempio: il LARP intitolato “Il Trono Nero 3: L’ascesa al potere” ha come eventi associati quello a “Castelfranco Veneto” il 19-10-2022 e quello a “Villafranca Veronese” il 30-02-2023. Il personaggio “Albert La Valette”, ad esempio, è stato scelto da un partecipante dell’evento di Castelfranco, ma da nessuno di quello di Villafranca.

Il **LARP** è caratterizzato da un titolo, dalla storia a cui si riferisce e dal capitolo, dalla sinossi del LARP e dalla lista di personaggi.

Il LARP può avere un **Evento** o più associati, e ognuno di essi ha un luogo, una data di inizio, un’ora di inizio, una data di fine e un’ora di fine.

Il **Luogo** viene definito dal nome della struttura, dal tipo di struttura, dalla città e dalla provincia. Un luogo può essere all’aperto, come un parco, o al chiuso, come un castello.

La **Storia** invece è costituita dal titolo e dalla trama generale.

Ogni **Personaggio**, invece, è caratterizzato da un nome, cognome, sesso, professione, da relazioni (parentela o associative) e tipo di personaggio (giocante detto PG, o non giocante, detto PNG). Con personaggio giocante si intendono i personaggi attivi agli eventi della storia, quelli non giocanti invece sono da considerare di sfondo, come mercanti, artigiani, soldati di guardia,

Ogni **Giocatore** è caratterizzato dal codice fiscale, nome, cognome, indirizzo e-mail, numero di telefono, data di nascita, luogo di nascita e dal livello calcolato come numero di eventi partecipati. Ogni membro dello **Staff,** nonostante abbia gli stessi attributi, può essere o meno admin della piattaforma, scelta preclusa per i **Giocatori.**

La **Prenotazione**, riferita ad un solo evento, contiene informazioni riguardo il cliente, l’evento, la data di prenotazione, il personaggio scelto, il tipo di prenotazione (intera, ridotta, associato, …), la

modalità di prenotazione dei costumi ed eventuali note facoltative. Una volta effettuato, il tipo di prenotazione non potrà essere cambiato. La prenotazione fornirà informazioni riguardante eventuale pernottamenti (in una struttura associata o in tenda).
La prenotazione avrà anche un costo totale come somma delle opzioni scelte.

Glossario dei Termini

Termine	Descrizione	Collegamenti
LARP (Live Action Role Play)	Il concetto di una giocata di ruolo dal vivo basata su una storia	Evento, Capitolo, Personaggio
Evento	La realizzazione del LARP come evento	Luogo, LARP
Luogo	La struttura (all'aperto o al chiuso) dove si svolge l'evento relativo ad un LARP	Evento
Personaggio Giocante (PG)	Personaggio attivo del LARP, le cui azioni influenzeranno la storia.	Evento, Prenotazione, Personaggio
Personaggio Non Giocante (PNG)	Personaggio di sfondo della storia, con cui i PG interagiscono.	Evento, Prenotazione, Staff, Personaggio
Giocatore	Persona la quale parteciperà all'evento nelle vesti di un personaggio	Evento, Prenotazione, Livello
Livello	Caratteristica indicante del numero di eventi partecipati.	Giocatore
Staff	Membro dell'associazione partecipante come PNG.	Personaggio Non Giocante (PNG)
Prenotazione	Prenotazione effettuata da un giocatore per un determinato evento.	Giocatore, Evento, Personaggio, Noleggio, Pernottamento, Importo
Noleggio	Noleggio di un set di costumi e/o accessori da parte di un giocatore.	Prenotazione, Costume
Storia	La storia di riferimento di uno o più LARP, costituita da capitoli.	Capitoli, LARP

Operazioni

Le operazioni seguenti vengono effettuate giornalmente:

Operazione	Tipo	Frequenza
Visualizzazione dei dettagli di un evento	L	2500 / giorno
Visualizzazione dei personaggi disponibili per un evento	L	2500 / giorno
Visualizzazione dei dettagli di un personaggio	L	8000 / giorno
Inserimento di un nuovo utente	S	100 / giorno
Calcolo del numero di eventi partecipati da un utente	L	100 / giorno
Aggiornamento dei dettagli di un utente	S	100 / giorno
Inserimento di una nuova prenotazione	S	50 / giorno
Aggiunta di un extra post prenotazione	S	20 / giorno
Visualizzazione del costo totale della prenotazione	L	500 / giorno
Visualizzazione delle prenotazioni associate ad un evento	L	500 / giorno
Calcolo del ricavo di un evento	L	20 / giorno
Visualizzazione dei noleggi attivi	L	500 / giorno
Visualizzazione delle prenotazioni richieste	L	500 / giorno

Le operazioni seguenti vengono effettuate mensilmente:

Operazione	Tipo	Frequenza
Inserimento di una nuova storia	S	1 / mese
Inserimento di un nuovo capitolo	S	1 / mese
Inserimento di un nuovo LARP	S	1 / mese
Inserimento di un nuovo evento	S	3 / mese
Aggiornamento dei dettagli dell'evento	S	1 / mese
Inserimento di un nuovo personaggio	S	20 / mese
Aggiornamento dei dettagli del personaggio	S	5 / mese

Progettazione Concettuale

Lista Entità

Se non specificato diversamente, viene implicitamente inteso che gli attributi siano NOT NULL.

LARP		
Codice	varchar(6) primary key	Il codice del LARP composto da tre lettere identificative della Storia e il capitolo a tre cifre
Titolo	varchar (50)	Il titolo del LARP
Sceneggiatura	varchar (255)	La sceneggiatura del LARP

Storia		
Titolo	varchar (50) primary key	Il titolo della storia
Trama	varchar (255)	La trama generale della storia

Evento		
Edizione	serial primary key	Numero dell'edizione (incrementale) in riferimento al LARP
DataInizio	datetime	La data e ora di inizio dell'evento
DataFine	datetime	La data e ora di fine dell'evento

Luogo		
Nome	varchar (50) primary key	Nome della struttura
Via	varchar (50) primary key	Via della struttura
Civico	tinyint primary key	Civico della struttura
Città	varchar (50) primary key	Città dove si trova la struttura
CAP	tinyint	Codice Postale della struttura
Provincia	char (2)	Lettere identificative della provincia

Personaggi		
Codice	varchar (6) primary key	Identificativo del personaggio composto dall'identificativo della storia del LARP (3 lettere) e da un incrementale con al massimo tre cifre
Nome	varchar (50)	Il Nome del personaggio
Cognome	varchar (50)	Il Cognome del personaggio

Pseudonimo	varchar (50) NULL: non ha pseudonimo	Il nome d'arte o il soprannome del personaggio, opzionale
Sesso	enum	Il sesso del personaggio (M – Maschio, F – Femmina, N – Non definito)
Professione	varchar (100)	Professione del personaggio contestualizzato

Il Personaggio si divide in PG e PNG con generalizzazione totale.

PG

PNG

Utente		
CF	varchar(16) primary key	Codice fiscale dell'utente
Nome	varchar(50)	Nome dell'utente
Cognome	varchar(50)	Cognome dell'utente
Email	varchar(255)	E-mail dell'utente
Telefono	varchar(18)	Prefisso (+39) e numero di telefono
PrimoSoccorso	boolean	L'utente è infermiere, paramedico o soccorritore qualificato

L'Utente si divide in Giocatore e Staff con generalizzazione totale.

Giocatore		
Livello	enum	Il livello del giocatore dato dal numero di LARP partecipati: • Novizio (N): < 5 LARP • Principiante (P): > 5 e < 10 LARP • Avanzato (A): > 10 e < 15 LARP • Esperto (E): > 15 LARP

Staff		
Admin	boolean	Lo staff è admin o no della piattaforma

Prenotazione		
Evento	int primary key	Identificativo dell'evento relativo alla prenotazione
Personaggio	varchar (6) primary key	Identificativo del personaggio relativo alla prenotazione
DataPrenotazione	datetime	Data e ora della prenotazione
Importo	money	L'importo totale della prenotazione
Iscrizione		
Tipologia	enum primary key	La tipologia di iscrizione scelta (I – Intero, R – Ridotto, A – Associato)
Importo	money	L'importo del tipo di prenotazione scelto

Extra		
ID	int primary key	Identificativo dell'extra richiesto
Importo	money	L'importo dell'extra richiesto

L'Extra si divide in Noleggio e Prenotazione con generalizzazione parziale

Noleggio		
Scelta	enum NULL: Si considera il costume portato da sé	Tipo di noleggio scelto: • 00 – PNG (Fornito dallo staff) • 01 – Weapon: armi • A0 – Base: costume • B0 – Deluxe: costume e accessori • C0 – Armor: costume, accessori e armatura • A1 – Base + Weapon • B1 – Deluxe + Weapon • C1 – Armor + Weapon

Pernottamento		
Opzione	enum NULL: Si considera il pernottamento in tenda e pasti in autonomia	Tipo di pernottamento scelto: • MP – Mezza pensione • CP – Pensione completa • BB – Bed & Breakfast

Lista Relazioni

Relazione	**Entità**		**Attributi**
Posizione	Luogo (1, N) Un luogo può essere scelto per più eventi	Evento (1,1) Un evento si svolge in un solo luogo	
Concretizzazione	Evento (1,1) Un evento fa riferimento a un solo LARP	LARP (1, N) Si possono svolgere più eventi riguardo un LARP	
Composizione	LARP (1,1) Un LARP è basato su un capitolo di una sola storia	Storia (1, N) Una storia può riferirsi a più LARP	Capitolo [int]
Ruolo	Personaggio (1, N) Un personaggio fa parte di almeno un LARP	LARP (1, N) Un LARP ha almeno un personaggio	
Relazione	Personaggio (0, N) Viene considerata soltanto la relazione più importante tra due personaggi (amorosa, amicizia, inimicizia, …)	Personaggio (0, N) Viene considerata soltanto la relazione più importante tra due personaggi (amorosa, amicizia, inimicizia, …)	Tipo [enum] Note [varchar(255)]
Riferimento	Giocatore (0, N) Un giocatore può avere più prenotazioni, ma non dello stesso evento.	Prenotazione (1,1) Una prenotazione fa riferimento ad un solo giocatore	
Modalità	Prenotazione (1,1) Una prenotazione può avere solo un tipo di prenotazione	Iscrizione (0, N) Una modalità di iscrizione si può riferire a più prenotazioni	
Riservato	Personaggio (0, N) Un personaggio può essere assegnato a più prenotazioni, ma	Evento (0, N) Un evento può avere più prenotazioni, ma non con lo	Prenotazione (1,1) Una prenotazione fa riferimento ad un solo

	non dello stesso evento.	stesso personaggio.	personaggio e un solo evento.	
Assegnazione	Staff (0, N) Uno staff può aver assegnato uno o più PNG anche dello stesso evento	PNG (1, N) Un PNG può essere assegnato a più Staff ma non nello stesso evento	Evento (0, N) Un evento può avere più assegnazioni di personaggi	
Richiesta	Noleggio (0, N) Una modalità di noleggio può riferirsi a più prenotazioni	Prenotazione (0,1) Una prenotazione si può riferire ad una sola modalità di noleggio		
Domanda	Pernottamento (0, N) Una modalità di pernottamento può riferirsi a più prenotazioni	Prenotazione (0,1) Una prenotazione si può riferire ad una sola modalità di pernottamento		

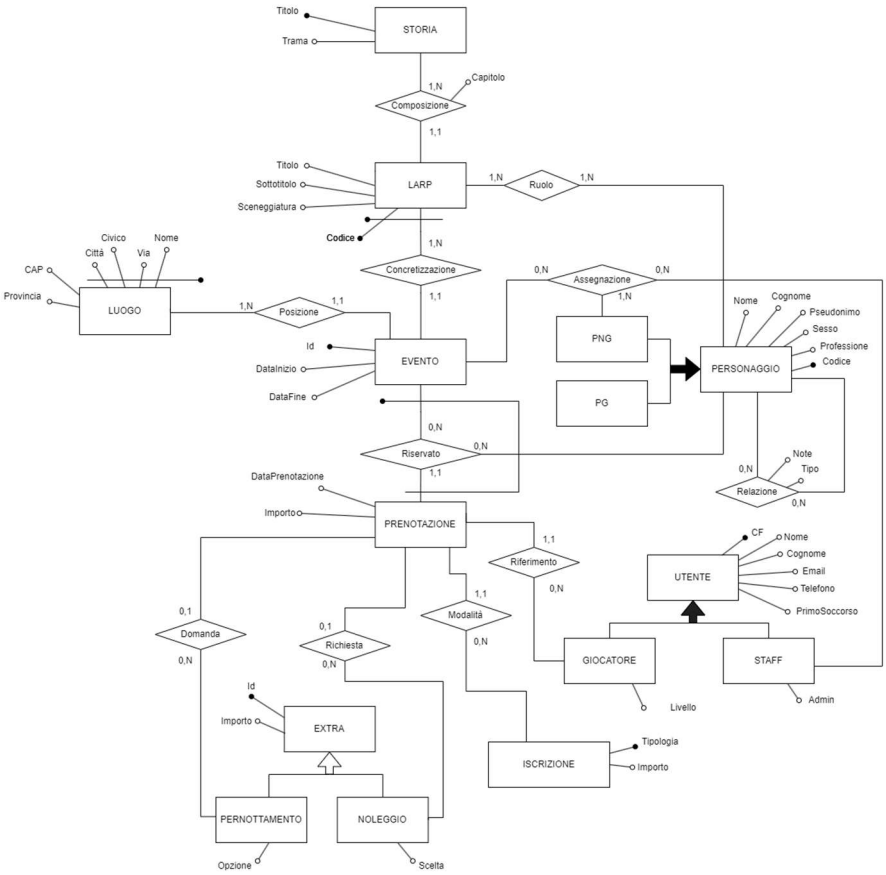
Vincoli non rappresentabili tramite schema E-R:

- Non possono esserci più prenotazioni effettuate dallo stesso utente sullo stesso evento ma con personaggi diversi.
- Non può esserci una prenotazione di un PNG assegnato ad un membro dello staff.
- Non si può modificare il tipo di iscrizione scelto ma si possono aggiungere e/o modificare gli extra entro il giorno prima della data di inizio dell'evento.
- Un giocatore può scegliere un Personaggio del suo stesso sesso o "non definito" (N) per motivi di trama.
- Possono avere delle relazioni soltanto dei personaggi dello stesso LARP
- Un personaggio non può avere relazioni con se stesso

Vincoli di derivazione:

- L'attributo Importo dell'entità Prenotazione è dato dalla somma degli extra e dal tipo di prenotazione scelto
- L'attributo Livello dell'entità Giocatore è dato dalla somma delle prenotazioni di eventi conclusi.

Schema Concettuale



Progettazione Logica

Analisi delle Ridondanze

L'Attributo **Importo** presente nell'entità **Prenotazione** può essere calcolato come somma dell'attributo Importo in Iscrizione ed Importo degli Extra relativi alla determinata Prenotazione. Tale ridondanza ha un peso sulle seguenti operazioni:

- Inserimento di una nuova prenotazione
- Aggiunta di un extra post prenotazione
- Visualizzazione del costo totale della prenotazione
- Calcolo del ricavo di un evento

Concetto	Tipo	Volume
Evento	E	100
Prenotazione	E	12000
Iscrizione	E	3
Noleggio	E	8
Pernottamento	E	3

Si è calcolata una media di al massimo 12000 prenotazioni suddivise per 100 eventi, ognuna con una scelta di iscrizione, noleggio e pernottamento scelta, 120 prenotazioni per un singolo evento.

Caso 1: Inserimento di una nuova prenotazione. (50 / giorno)

Consideriamo il caso dove, nella prenotazione effettuata, vengano scelte una modalità di noleggio e una di pernottamento.

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Noleggio	E	1	L
Pernottamento	E	1	L
Iscrizione	E	1	L
Prenotazione	E	1	S

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Prenotazione	E	1	S

Caso 2: Aggiunta di un extra post prenotazione. (20 / giorno)

Sia nel caso di noleggio che nel caso di pernottamento il procedimento sarebbe lo stesso. Viene considerato noleggio quindi puramente per comodità.

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Noleggio	E	1	L
Pernottamento	E	1	L
Iscrizione	E	1	L
Prenotazione	E	1	S

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Noleggio	E	1	S

Caso 3: Visualizzazione del costo totale della prenotazione. (50 / giorno)

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Prenotazione	E	1	L

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Noleggio	E	1	L
Pernottamento	E	1	L
Iscrizione	E	1	L

Caso 4: Calcolo del ricavo di un evento (20 / giorno)

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Evento	E	1	L
Prenotazione	E	120	L

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Evento	E	1	L
Prenotazione	E	120	L
Iscrizione	E	120	L
Noleggio	E	120	L
Pernottamento	E	120	L

Calcolo del Costo Giornaliero:

Tipo Ridondanza	Costo Giornaliero
Presente	Caso 1: 50 + 50 + 50 + 50 * 2 = 250 Caso 2: 20 + 20 + 20 + 20 * 2 = 140 Caso 3: 50 Caso 4: 20 + 20 * 120 = 2420 Totale: 2860
Assente	Caso 1: 50 * 2 = 100 Caso 2: 20 * 2 = 40 Caso 3: 50 + 50 + 50 = 150 Caso 4: 20 + 20 * 120 + 20 * 120 + 20 * 120 + 20 * 120 = 9620 Totale: 9910

Conclusione

In conclusione, si può notare come le operazioni di visualizzazione siano più incisive per il calcolo del costo di mantenimento dell'attributo **Importo** di **Prenotazioni**, a discapito di quelle di modifica. Esse dimostrano palesemente quanto il mantenimento della ridondanza sia importante per la riduzione dei costi delle operazioni giornaliere. Dato ciò, l'attributo verrà mantenuto ed aggiornato ad ogni inserimento e modifica delle modalità di pernottamento, nonostante l'alto rischio di inconsistenza. Viene considerato inoltre che l'attributo **Importo** (money) occupa 8 bytes, moltiplicato per il numero di prenotazioni (12000), per un totale di 96 KB di spazio occupato. La rimozione dell'attributo per dimensioni eccessive viene quindi considerata non necessaria.

L'Attributo **Livello** presente nell'entità **Giocatore** può essere calcolato come somma delle prenotazioni di eventi conclusi. Come eventi conclusi vengono considerati gli eventi il cui attributo DataFine è antecedente la data corrente. Le operazioni coinvolte quindi sono:

- Inserimento di un nuovo utente
- Calcolo del numero di eventi partecipati da un utente
- Aggiornamento dei dettagli di un utente

Viene considerata nuovamente l'ipotesi di 100 eventi, come enunciato precedentemente. Inoltre, viene considerata l'ipotesi che ci siano 2000 utenti, divisi tra 1900 giocatori e 100 membri dello staff. Si considera una media di circa 6 prenotazioni per giocatore (12000 prenotazioni / 1900 giocatori), ricordando che un membro dello staff non effettua prenotazioni.

Concetto	Tipo	Volume
Evento	E	100
Prenotazione	E	12000
Utente	E	2000

Caso 1: Inserimento di un nuovo utente. (100 / giorno)

Indipendentemente dall'assenza o presenza della ridondanza dell'attributo, verrebbe fatta in qualsiasi caso soltanto un'operazione di scrittura su Utente, dato che nel momento stesso dell'inserimento di un nuovo utente, esso avrebbe sempre associato 0 prenotazioni. Questa operazione, dunque, è superflua per il calcolo del costo giornaliero.

Caso 2: Calcolo del numero di eventi partecipati da un utente (100 / giorno)

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Utente	E	1	L

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Utente	E	1	L
Prenotazione	E	6	L
Evento	E	6	L

Caso 3: Aggiornamento dei dettagli di un utente (100 / giorno)

Ridondanza presente:

Concetto	Tipo	Accessi	Tipo
Prenotazione	E	6	L
Evento	E	6	L
Utente	E	1	S

Ridondanza assente:

Concetto	Tipo	Accessi	Tipo
Utente	E	1	S

Calcolo del Costo Giornaliero:

Tipo Ridondanza	Costo Giornaliero
Presente	Caso 2: 100 Caso 3: 100 * 6 + 100 * 6 + 100 * 2 = 1500 Totale: 1500
Assente	Caso 2: 100 + 100 * 6 + 100 * 6 = 1300 Caso 3: 100 * 2 Totale: 1500

Conclusione

In conclusione, dato che le operazioni che riguardano l'entità **Utente** sono simili tra loro, il costo giornaliero è medesimo. Quindi, dato che comporta spazio di memoria inutilmente occupato e rischio di inconsistenza, viene eliminato.

Eliminazione delle Generalizzazioni

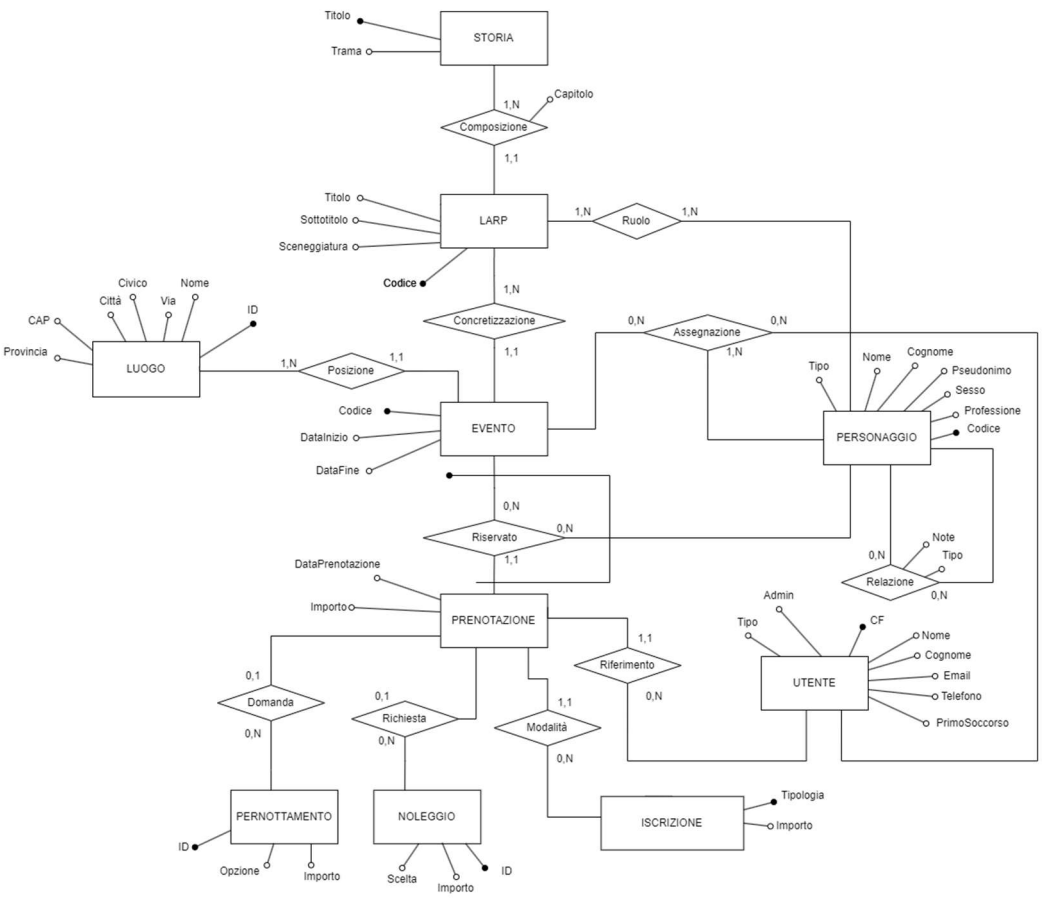
Generalizzazione	Risoluzione
Utente ⇐ Giocatore, Staff	Le entità Giocatore e Staff vengono accorpate in Utente, dato l'alto numero degli attributi comuni ad entrambe. • Tipo [enum]: Specifica il tipo di utente (G – Giocatore, S - Staff) • Livello assume valore NULL per gli utenti di tipo staff • Admin assume valore NULL per gli utenti di tipo giocatore
Personaggio ⇐ PG, PNG	Le entità PG e PNG vengono accorpate all'entità genitore • Tipo [enum]: Specifica il tipo di personaggio (PG/PNG)
Extra ← Noleggio, Pernottamento	Le entità Noleggio e Pernottamento vengono separate in due entità diverse per diversità negli enumerati e per mantenere il concetto di un solo noleggio e pernottamento per prenotazione. • ID relativo ad Extra viene aggiunto alle due entità e rimane chiave primaria • Importo viene aggiunto alle due entità

Scelta degli Identificatori Primari

Il salvataggio di tutti gli attributi della chiave primaria di **Luogo** all'interno di Evento risulta poco pratico. Dunque, viene introdotto un **ID** auto incrementale all'interno dell'entità **Luogo** come nuova chiave primaria, collegata poi ad Evento.

Inoltre, la chiave primaria di Evento (**Edizione, LARP**) viene sostituita da una chiave unica chiamata **Codice.** Conterrà il codice del LARP e l'ultima cifra indicherà l'edizione. Come esempio, considerando il LARP "Il Trono Nero", capitolo 3, che, come codice, ha "TRN3", l'evento come quarta edizione del LARP descritto sarà indicato "TRN3_4".

Schema Ristrutturato



Schema Relazionale

Sotto la relazione vengono elencate le chiavi esterni con i loro riferimenti. Gli attributi con l'asterisco, invece, possono accettare valori nulli.

Storia (Titolo, Trama)
LARP (Codice, Storia, Titolo, Capitolo, Sceneggiatura)
- LARP.Storia → Storia.Titolo

Personaggio (Codice, Tipo, Nome, Cognome, Pseudonimo*, Sesso, Professione)
Relazione (Persona1, Persona2, Tipo, Note)
- Relazione.Persona1 → Personaggio.Codice
- Relazione.Persona2 → Personaggio.Codice

Ruolo (LARP, Personaggio)
- Ruolo.LARP → LARP.Codice
- Ruolo.Personaggio → Personaggio.Codice

Luogo (ID, Nome, Via, Civico, Città, CAP, Provincia)
Utente (CF, Nome, Cognome, Email, Telefono, PrimoSoccorso, Tipo, Admin*)
Pernottamento (ID, Opzione, Importo)
Noleggio (ID, Scelta, Importo)
Iscrizione (Tipologia, Importo)
Evento (Codice, LARP, Luogo, DataInizio, DataFine)
- Evento.LARP → LARP.Codice
- Evento.Luogo → Luogo.ID

Prenotazione (Evento, Personaggio, Utente, DataPrenotazione, Importo, Iscrizione, Noleggio*, Pernottamento*)

- Prenotazione.Evento → Evento.Codice
- Prenotazione.Personaggio → Personaggio.Codice
- Prenotazione.Utente → Utente.CF
- Prenotazione.Iscrizione → Iscrizione.Tipologia
- Prenotazione.Noleggio → Noleggio.ID
- Prenotazione.Pernottamento → Pernottamento.ID

Assegnazione (Evento, Personaggio, Utente)

- Assegnazione.Evento → Evento.Codice
- Assegnazione.Personaggio → Personaggio.Codice
- Assegnazione.Utente → Utente.CF

Query e Indici

Indici

Dato che la visualizzazione di un evento è estremamente più frequente rispetto all'inserimento di uno nuovo, risulta utile la visualizzazione degli eventi in ordine di data, così che possano apparire prima quelli prossimi. Si è deciso quindi di indicizzare codice, datainizio e datafine nell'indice descritto come Eventi_date.

Query

1. **Visualizzare il numero di prenotazioni di ogni evento associato ad un determinato LARP.** (nell'esempio con codice 'HDL')

```
SELECT pr.evento, count(*) as conteggio
FROM Prenotazione as pr
WHERE pr.evento LIKE 'HDL%'
GROUP BY pr.evento
ORDER BY pr.evento ASC
```

	evento character varying (9)	conteggio bigint
1	HDL1_1	1
2	HDL2_1	9
3	HDL2_2	6
4	HDL2_3	1

2. **Visualizzare tutti i personaggi non ancora assegnati o prenotati di uno specifico evento.** (nell'esempio con codice 'SHO3_1)

```
CREATE VIEW prenotazioni_personaggi AS
SELECT pr.utente, pr.evento, pr.personaggio FROM Prenotazione as pr
JOIN evento as e on e.codice=pr.evento;

CREATE VIEW assegnazioni_personaggi AS
SELECT asg.utente, asg.evento, asg.personaggio FROM Assegnazione as asg
JOIN evento as e on e.codice=asg.evento;

SELECT pg.* as evento from Personaggio as pg
JOIN Ruolo as r on r.personaggio = pg.codice
JOIN larp as lp on lp.codice = r.larp
JOIN evento as e on e.larp = lp.codice
WHERE e.codice LIKE 'SHO3_1'
AND pg.codice NOT IN (
    SELECT pr.personaggio from prenotazioni_personaggi as pr
    WHERE pr.evento LIKE 'SHO3_1'
)
AND pg.codice NOT IN (
    SELECT ap.personaggio from assegnazioni_personaggi as ap
    WHERE ap.evento LIKE 'SHO3_1'
)
```

	codice [PK] character varying (6)	nome character varying (50)	cognome character varying (50)	pseudonimo character varying (50)	sesso sex	professione character varying
1	SHO1	Zachary	Windler	vainly	N	Studente
2	SHO10	Ralph	Watsica	cleverly	N	Bidello
3	SHO11	Roman	Bailey	strictly	N	Professore
4	SHO12	Julie	Reichel	[null]	F	Studente
5	SHO14	Brendan	Dicki	thoroughly	N	Studente
6	SHO16	Mack	Pfeffer	[null]	N	Studente
7	SHO19	Daniel	Muller	scarcely	N	Professore
8	SHO2	Bonnie	Bartell	commonly	F	Studente
9	SHO22	Johnny	Kub	[null]	N	Professore
10	SHO24	Kyle	Wunsch	[null]	M	Bidello
11	SHO27	Wendell	Heller	yieldingly	N	Studente
12	SHO28	Conrad	Sauer	helplessly	M	Studente

3. **Visualizzare il luogo dove si sono svolti il maggior numero di eventi.**

```
DROP VIEW luoghi_eventi;

CREATE VIEW luoghi_eventi AS
SELECT l.nome, COUNT(*) AS totale
FROM Luogo as l
JOIN evento as e ON e.luogo=l.id
GROUP BY l.nome
ORDER BY totale DESC, l.nome DESC;

SELECT * FROM luoghi_eventi as le
limit 1;
```

	nome character varying (50)	totale bigint
1	Palazzetto dello Sport...	12

4. **Visualizzare quante volte è stato scelto un determinato personaggio tra prenotazioni e assegnazioni.** (nell'esempio il personaggio 'IGN7')

```
SELECT storia.titolo, COUNT(pr.evento) as conteggio,
SUM(pr.importo) as ricavo FROM storia
JOIN larp ON larp.storia = storia.titolo
JOIN evento as e ON e.larp = larp.codice
JOIN prenotazione as pr ON pr.evento = e.codice
GROUP BY storia.titolo
```

	codice character varying (6)	count bigint
1	IGN7	6

5. **Visualizzare i titoli di ogni storia, il numero di prenotazioni di tutti gli eventi a loro associati e il loro ricavo totale.**

```
SELECT pgtotal.codice, count(pgtotal.codice)
FROM(
    SELECT pg.codice FROM personaggio as pg
    JOIN prenotazione as pr ON pr.personaggio = pg.codice
    UNION ALL
    SELECT pg1.codice from personaggio as pg1
    JOIN assegnazione as asg ON asg.personaggio = pg1.codice
) as pgtotal
GROUP BY pgtotal.codice
HAVING pgtotal.codice LIKE 'IGN7'
```

	titolo [PK] character varying (50)	conteggio bigint	ricavo money
1	Fatti amare	1	406,54 €
2	Voglio crederci	11	1.700,14 €
3	Hello Dolly	17	2.645,08 €
4	I ragazzi d'estate	9	1.064,15 €
5	Stormy Weather (Keeps R...	16	3.180,30 €
6	Incenso e Menta Piperita	13	1.542,78 €
7	School's Out	49	7.912,10 €
8	In ginocchio	37	7.189,14 €
9	Noi siamo il Mondo	9	2.163,48 €

6. **Visualizzare, in base al codice fiscale di un giocatore, i personaggi di altri giocatori iscritti allo stesso evento con cui il personaggio prenotato ha delle interazioni, e il tipo**

```
DROP VIEW prenotazioni_personaggi;

CREATE VIEW prenotazioni_personaggi AS
SELECT pr.utente, pr.evento, pr.personaggio FROM Prenotazione as pr
JOIN evento as e on e.codice=pr.evento;

SELECT pp1.utente, pp1.evento, pp1.personaggio, r1.personaggio2
as interazione, r1.tipo as relazione
FROM prenotazioni_personaggi as pp1
JOIN Relazione as r1 on r1.personaggio1 = pp1.personaggio
WHERE pp1.utente LIKE 'DSNLJS2NL3LJ5V1T'
AND r1.personaggio2 IN (
    SELECT pp2.personaggio FROM prenotazioni_personaggi as pp2 WHERE
    pp2.personaggio = r1.personaggio2
    AND pp2.evento = pp1.evento
)
UNION
SELECT pp3.utente, pp3.evento, pp3.personaggio, r2.personaggio1
as interazione, r2.tipo as relazione
FROM prenotazioni_personaggi as pp3
JOIN Relazione as r2 on r2.personaggio2 = pp3.personaggio
WHERE pp3.utente LIKE 'DSNLJS2NL3LJ5V1T'
AND r2.personaggio1 IN (
    SELECT pp4.personaggio from prenotazioni_personaggi as pp4 WHERE
    pp4.personaggio = r2.personaggio1
    AND pp4.evento = pp3.evento
)
```

	utente character varying (16)	evento character varying (9)	personaggio character varying (6)	interazione character varying (6)	relazione relationship
1	DSNLJS2NL3LJ5V1T	VCR1_1	VCR32	VCR12	Lavorativo
2	DSNLJS2NL3LJ5V1T	BSH1_2	BSH5	BSH22	Amicizia
3	DSNLJS2NL3LJ5V1T	SHO2_1	SHO40	SHO4	Familiare
4	DSNLJS2NL3LJ5V1T	BSH3_3	BSH23	BSH33	Amicizia
5	DSNLJS2NL3LJ5V1T	SHO3_1	SHO3	SHO9	Familiare
6	DSNLJS2NL3LJ5V1T	SHO1_3	SHO41	SHO11	Amoroso
7	DSNLJS2NL3LJ5V1T	BSH1_3	BSH4	BSH5	Amicizia
8	DSNLJS2NL3LJ5V1T	RGT1_2	RGT24	RGT15	Inimicizia
9	DSNLJS2NL3LJ5V1T	BSH1_3	BSH4	BSH32	Lavorativo

Codice (C++)

Setup

Per il corretto funzionamento del codice, è necessario inserire le dipendenze del programma relative a PostgreSQL all'interno della cartella "dependencies", come descritto nel documento "Accesso a PostgreSQL da software" presentato nel Laboratorio 7 del corso.

Compilazione

È necessario lanciare il comando di compilazione per la creazione dell'eseguibile:

g++ -static-libstdc++ leonerampante.cpp -L dependencies\lib -lpq -o [nome_eseguibile]

[nome_eseguibile] verrà sostituito con il nome effettivo dell'eseguibile (senza parentesi quadre).

Esecuzione

Tramite prompt comandi, lanciare l'eseguibile. Verranno richiesti, in ordine, l'utente, la password, il database di riferimento, l'indirizzo IP e la porta dell'host relativi a PostgreSQL. Viene considerato implicito il caricamento dei dati presenti nel file "LeoneSQL.sql" all'interno del database.
Se la connessione è avvenuta con successo, verrà mostrato un piccolo menu dove l'utente, tramite inserimento da tastiera, potrà scegliere quale query eseguire, se stampare la lista descrittiva delle query, o uscire dal programma.
Per le query parametriche verrà inoltre richiesto il determinato parametro sempre tramite inserimento da tastiera. Successivamente verrà eseguita la query scelta e l'utente verrà portato al menù precedente.

Codice

L'esecuzione avviene all'interno del main. Vengono richiamate determinate funzioni, ora descritte, per la facilità di lettura del codice e per comodità.

getConnectionParameters: viene eseguita per ogni parametro necessario, e riceve in input il nome del parametro da mostrare in input.

```
char* getConnectionParameters(string param) {
```

connect: riceve come parametri della funzione quelli inseriti dall'utente, per poi effettuare il collegamento al database tramite PostgreSQL

```
PGconn* connect(const char* user, const char* pwd, const char* db, const char* host, const char* port) {
```

execute: dopo che l'utente sceglie la query e viene "parametrizzata", viene eseguita appoggiandosi alla connessione precedentemente stabilita.

```
PGresult* execute(PGconn* conn, const char* query) {
```

printResult: ottiene il risultato dalla funzione **execute** e stampa le intestazioni e le tuple relative.

```
void printResult(PGresult* result) {
```

printQueryList: funzione utilitaria per stampare la lista descrittiva delle query

```
void printQueryList(){
```

choseQuery: funzione principale del programma, che si occupa del funzionamento del menu a

```
char* choseQuery(PGconn* conn){
```

scelta e dell'ottenimento dei parametri per le query parametrizzate. Si occupa anche dell'interruzione della connessione in caso l'utente richieda di farlo.