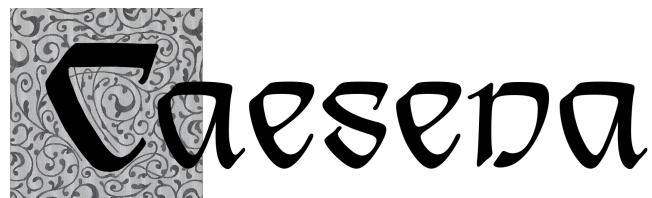


Caesena

Elaborato per l'esame "Basi di dati"



Mauro Pellonara (matricola 0001020678)
Davide Speziali (matricola 0001027635)

Basi di dati

Università di Bologna

27/06/2023

Indice

1 Analisi dei requisiti	2
1.1 Intervista	2
1.2 Estrazione dei concetti principali	3
2 Progettazione Concettuale	6
2.1 Schema scheletro	6
2.2 Schema concettuale finale	8
3 Progettazione logica	10
3.1 Stima del volume dei dati	10
3.2 Descrizione delle operazioni principali e stima della frequenza .	11
3.3 Schemi di navigazione e tabelle degli accessi	12
3.4 Raffinamento dello schema	19
3.5 Analisi delle ridondanze	21
3.6 Traduzione di entità e associazioni in relazioni	22
3.7 Schema relazionale finale	24
3.8 Traduzione delle operazioni in query SQL	27
4 Progettazione dell'applicazione	33
4.1 Descrizione dell'architettura	33
Riferimenti bibliografici	36

1 Analisi dei requisiti

1.1 Intervista

Si vuole realizzare la trasposizione videoludica del gioco da tavolo Carcassonne [1] nel quale i giocatori devono posizionare tessere e seguaci per formare strutture e realizzare punti. Chi alla fine della partita avrà totalizzato più punti verrà proclamato vincitore.

Ogni giocatore sarà identificato univocamente dal nome, potrà partecipare a più partite diverse contemporaneamente e avrà un colore e un punteggio distinto per ognuna di esse.

Ogni partita dovrà permettere l'uso di più espansioni, ovvero insiemi di diverse tessere, strutture e seguaci. Inoltre, si potrà mettere in pausa una partita e riprenderla in un secondo momento.

Le tessere hanno un ordine preciso nella quale devono esse giocate e sono di diversi tipi, ogni tipo è contraddistinto dall'espansione alla quale appartiene e da quante tessere di quel tipo vanno usate in una partita. Inoltre, prima di essere posizionata sul campo ogni tessera può essere ruotata.

All'interno di una tessera sono presenti più sezioni, contraddistinte dalla struttura e dall'eventuale seguace su di loro presente. Ognuna di queste dovrà poter essere considerata chiusa, ovvero combaciante con un'altra sezione di una tessera adiacente. È quindi necessario per ogni tessera definire un modo per poterla configurare alla sua creazione.

Le strutture sono di diversi tipi e possono contenere dei punti ed essere considerate chiuse. Ogni tipo è contraddistinto dall'espansione alla quale appartiene, da quanti punti una struttura di quel tipo contiene alla creazione e dal numero per cui dividere i punti rimasti a fine partita.

I seguaci possono essere piazzati su una sezione di una tessera, sono di diversi tipi ed appartengono ad un giocatore. Ogni tipo è contraddistinto dalla propria forza, dall'espansione alla quale appartiene e da quanti seguaci di quel tipo vanno assegnati ad ogni giocatore.

Ulteriormente, in vista della possibile aggiunta futura di una modalità online, bisognerà permettere ai giocatori di scegliere la regione e il server nella quale giocare. È quindi necessario tenere traccia di quante partite sono giocate su uno stesso server e se questo è acceso o spento.



Figura 1: Esempio di una struttura di tipo Città con sopra un seguace.

1.2 Estrazione dei concetti principali

Termino	Breve descrizione	Eventuale sinonimi
Giocatore	Persona che partecipa a diverse partite, è in grado di ottenere punti dal piazzamento di seguaci	Player, Utente
Partita	Istanza di gioco, termina se tutte le tessere sono state piazzate. Al termine viene decretato il giocatore vincitore	Game
Espansione	Insieme di tessere, strutture e seguaci aggiuntivi	Expansion
Tessera	Elemento di gioco da piazzare a ogni turno della partita in modo tale che combaci con le tessere precedenti	Tassello, tile
Sezione	Parte di una tessera contenente una struttura sulla quale è possibile piazzare un seguace	Section
Struttura	Componente di gioco che permette di ricevere punti in caso di presenza di un seguace	Gameset
Seguace	Componente di gioco posizionabile nelle strutture, alla chiusura di esse il seguace viene restituito al giocatore	Pedina, Meeple
Server	Sistema o piattaforma che ospita le partite online	
Regione	Area geografica o regionale nella quale giocare	Region, area

Dopo aver letto e compreso i requisiti, si procede a scrivere un documento che riassume tutte le idee e, in particolare, evidenzi i concetti principali eliminando eventuali ambiguità.

Ogni **giocatore**, identificato univocamente con un nome, deve essere in grado di partecipare a più partite, anche contemporaneamente. Sono presenti informazioni legate al giocatore che valgono solo all'interno di una **partita**, in particolar modo il punteggio e il **colore**, per questo motivo diventa necessario considerare l'esistenza di un **giocatore in partita**.

Inoltre per ogni partita bisogna salvarne lo stato di completamento. Ogni partita può fare uso di diverse **espansioni**, che possono contenere nuovi seguaci, nuove tessere e nuove strutture. Inoltre per ogni partita devono essere univocalmente presenti un determinato numero di seguaci, tessere e strutture, che non possono quindi esistere fuori dal contesto di una partita.

Per ogni **seguace** bisogna sapere il giocatore proprietario, il tipo di seguace e se è correntemente piazzato all'interno di una sezione.

Il **tipo di seguace** esiste al di fuori del contesto della partita e determina, di un dato seguace, l'espansione di cui fa parte, la sua forza e la quantità di seguaci di suddetto tipo che vanno assegnati a ogni giocatore.

Di ogni **tessera** è necessario salvare l'ordine all'interno della partita, la posizione sul tabellone e il tipo. Ulteriormente, una tessera è composta anche da diverse **sezioni**. Ognuna di queste contiene una **struttura** ed ha un certo **tipo di sezione** che determina le sezioni circostanti. È anche necessario avere un **configuratore di tipo di tile** che definisce la correlazione di un tipo di tile, il tipo delle sezioni e il tipo delle strutture presenti su di esse.

Il **tipo di tessera** esiste al di fuori del contesto della partita e determina, sulla base di una data tessera, l'espansione di cui fa parte e la quantità di tessere di quel tipo che devono essere presenti in una partita.

Di ogni struttura bisogna sapere se è chiusa o meno e il punteggio. Le strutture possono essere di diversi tipi e il **tipo di struttura**, che esiste al di fuori del contesto di una partita, indica l'espansione alla quale appartiene, quanti punti una struttura di quel tipo contiene alla creazione e il numero per cui dividere i punti rimasti a fine partita.

Le partite sono giocate all'interno di un **server** e per ognuno se ne memorizza lo stato (se è attivo o meno) e il numero massimo di partite giocabili. Per ogni server bisogna anche sapere la **regione** in cui si trova, che è composta dal **continente** e dal **punto cardinale**.

Segue per chiarezza quali elementi esistono nel contesto di una partita e quali invece sono universali:

Elemento	Univoco ad una partita
Struttura	Si
Giocatore in partita	Si
Seguace	Si
Settore	Si
Tessera	Si
Giocatore	No
Espansioni	No
Tipo di seguace	No
Tipo di struttura	No
Tipo di tessera	No
Tipo di sezione	No
Server e regione	No
Configuratore di tipo di tile	No

Si conclude con una lista delle principali azioni richieste:

- Creare una nuova partita
- Entrare in una partita in corso
- Vedere delle statistiche globali

2 Progettazione Concettuale

2.1 Schema scheletro

Un giocatore deve poter giocare a più partite contemporaneamente, quindi una semplice associazione tra giocatore e partita non è sufficiente ed è necessario avere un'entità PlayerInGame che contenga le informazioni del giocatore riguardanti una determinata partita. Inoltre, quest'ultima dovrà essere collegata con l'entità Color. Non potendo nella stessa partita avere più giocatori con lo stesso colore si è reso necessario evitare che esso fosse parte della chiave primaria del giocatore.

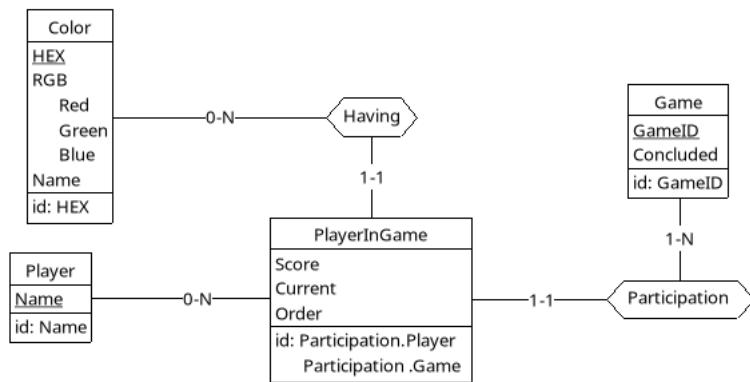


Figura 2: Schema scheletro riguardante giocatori, colori e game

Le entità Gameset, Tile e Meeple seguono un simile ragionamento riguardo al loro rapporto con una partita. Ognuna di queste entità ha un collegamento con il proprio "tipo": rispettivamente le entità GamesetType, TileType e MeepleType. Per tutti e tre viene fatta la distinzione tra le versioni presenti nel gioco base e quelle presenti nelle espansioni, è quindi chiaramente presente una generalizzazione che si divide in Basic (BasicGameset, BasicTile, BasicMeeple) e in Expansion (ExpansionGameset, ExpansionTile, ExpansionMeeple). Quest'ultima deve ovviamente essere connessa ad Expansion.

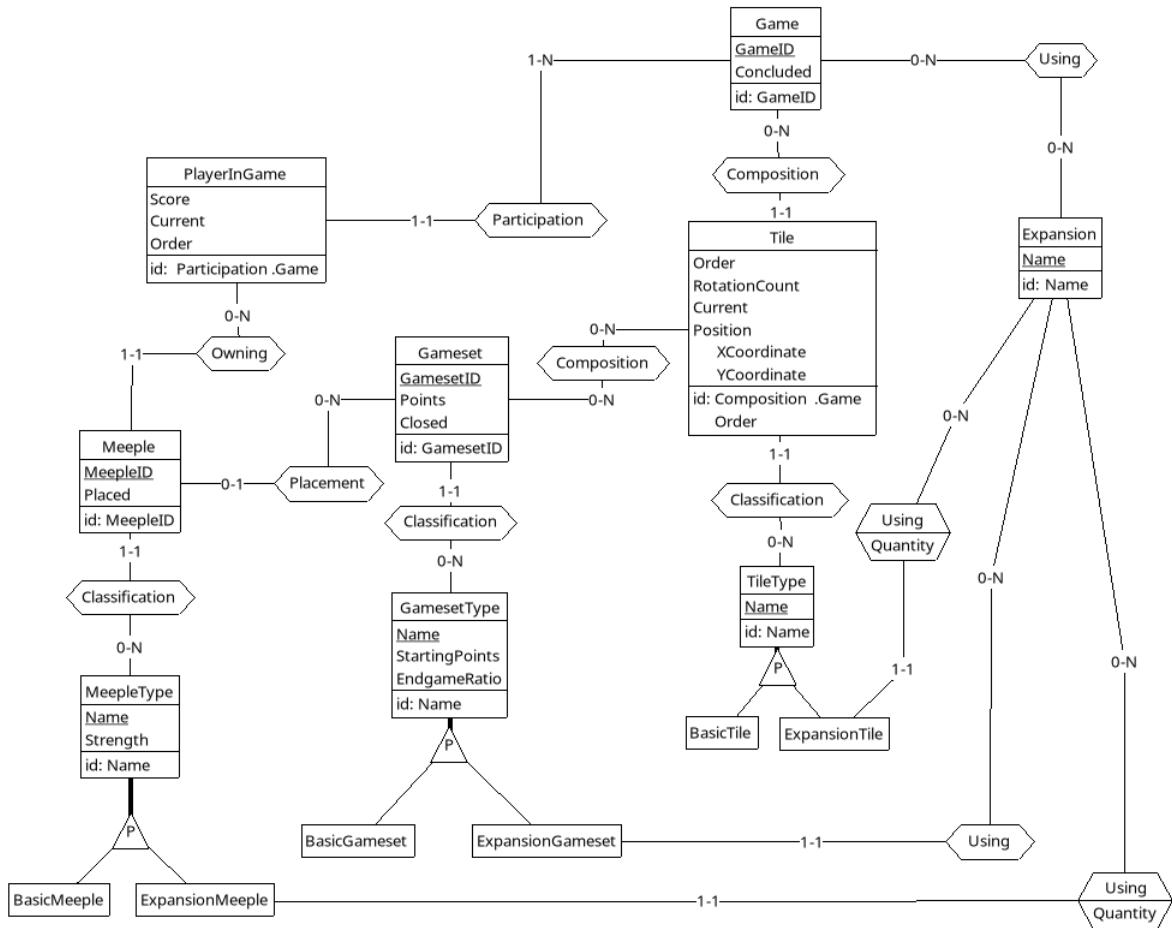


Figura 3: Schema scheletro riguardante strutture, tessere, meeple e espansioni

È necessario un modo per generare correttamente all'inizio della partita le Tile secondo il proprio tipo, a questo scopo si utilizza l'entità TileTypeConfiguration. Esso, per ogni TileType, contiene le informazioni riguardanti i tipi di sezione (entità TileSectionType) e i tipi di strutture (entità GamesetType) che dovranno essere presenti su tale tessera. Sulla base di GamesetType vengono poi creati Gameset che in combinazione con Tile e TileSectionType definiscono una TileSection. Su una TileSection può essere piazzato un Meeple e può essere ritenuta chiusa se combaciante con un'altra TileSection.

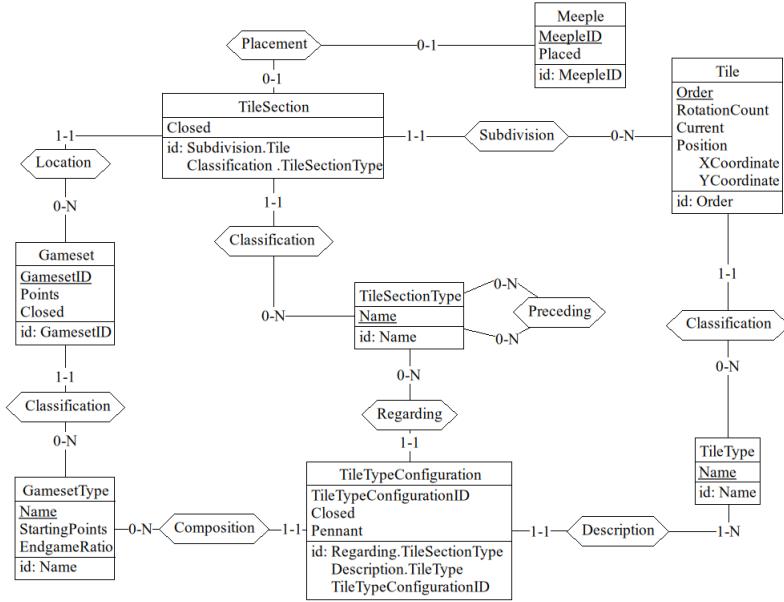


Figura 4: Schema scheletro riguardante la configurazione di tessere

Ogni partita deve essere ospitata all'interno di un Server e ogni Server deve far parte di una Region, che è a sua volta definita come la combinazione tra un Continent e un punto CardinalPoint.

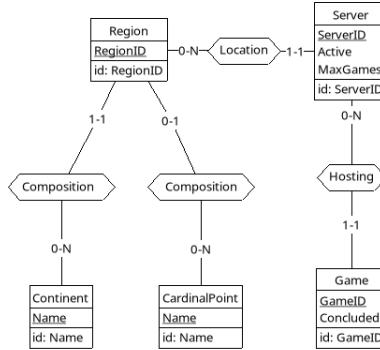
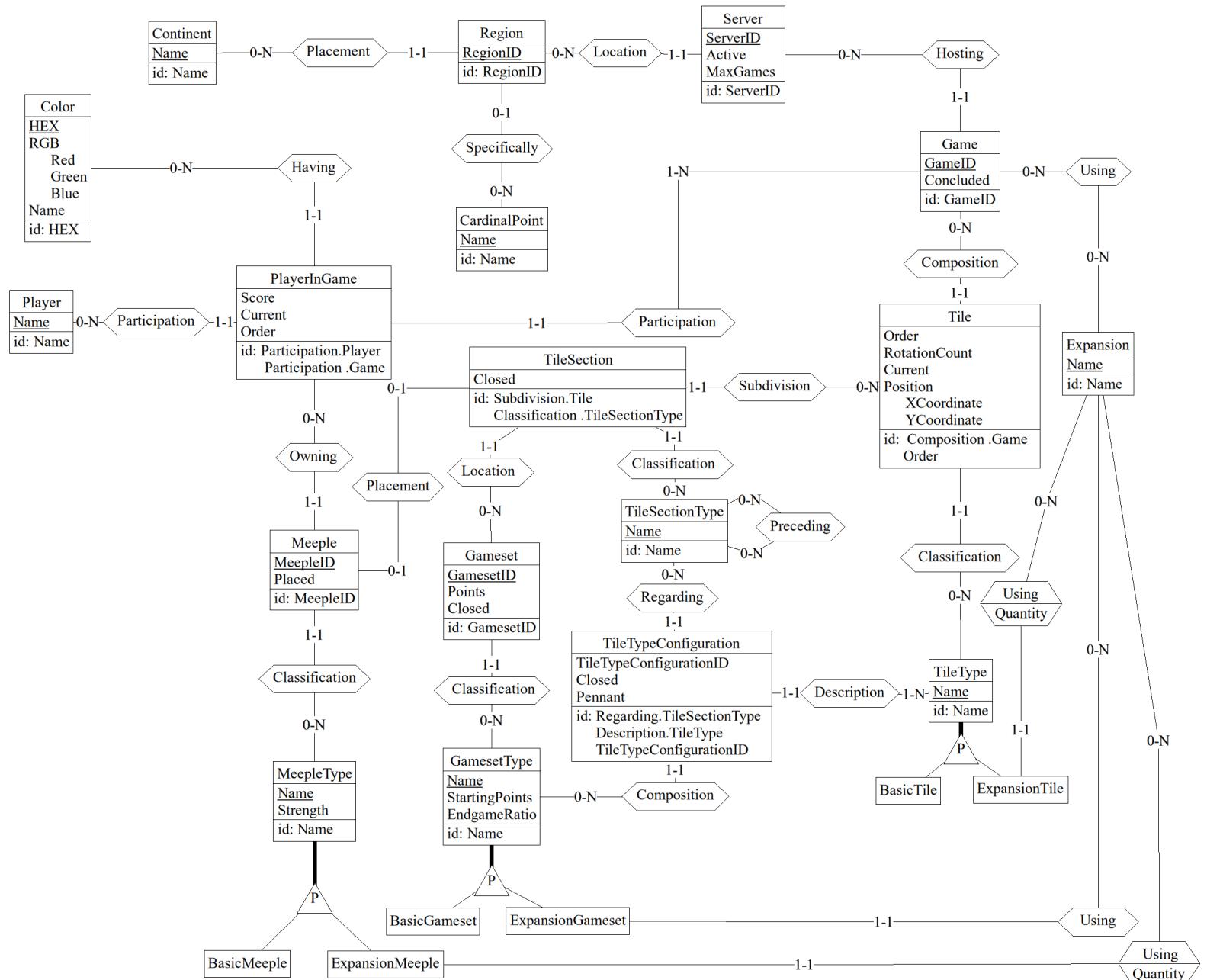


Figura 5: Schema scheletro riguardante la gestione dei server

2.2 Schema concettuale finale



3 Progettazione logica

3.1 Stima del volume dei dati

Si suppone che siano state giocate 10 partite alla quale hanno partecipato ogni volta 4 giocatori diversi. Le partite prevedono mediamente 70 tessere in totale e 8 seguaci per ogni giocatore.

Concetto	Tipo	Volume	Concetto	Tipo	Volume
Player	E	40	Tile	E	700
Participation	R	40	Composition	R	700
PlayerInGame	E	40	Classification	R	700
Having	R	40	BasicTile	E	18
Color	E	10	ExpansionTile	E	7
Meeple	E	320	Using	R	7
Owning	R	320	TileSection	E	9.100
Classification	R	320	Subdivision	R	9.100
BasicMeeple	E	1	Location	R	9.100
ExpansionMeeple	E	1	Classification	R	9.100
Using	R	1	Placement	R	200
Gameset	E	1.500	TileSectionType	E	13
Classification	R	1.500	Preceding	R	12
BasicGameset	E	5	TileTypeConfiguration	E	325
ExpansionGameset	E	0	Description	R	325
Using	R	0	Regarding	R	325
Game	E	10	Composition	R	325
Participation	R	40			
Hosting	R	10			
Using	R	10			
Expansion	E	4			
Server	E	10			
Location	R	10			
Region	E	6			
Placement	R	6			
Continent	E	4			
Specifically	R	6			
CardinalPoint	E	5			

3.2 Descrizione delle operazioni principali e stima della frequenza

Si suppone che ogni giorno vengano giocate 10 partite alla quale partecipano ogni volta 4 giocatori diversi. Le partite vengono sempre concluse nell'arco della giornata e si prevede che mediamente vengono giocate 70 tessere a partite.

Cod.	Operazione	Frequenza
1	Creare una nuova partita	10 al giorno
2	Visualizzare i server di gioco disponibili	10 al giorno
3	Visualizzare tutte le espansioni	10 al giorno
4	Creare un nuovo giocatore	40 al giorno
5	Visualizzare tutti i colori	40 al giorno
6	Visualizzare il giocatore corrente	700 al giorno
7	Visualizzare la tessera corrente	700 al giorno
8	Ruotare la tessera corrente	300 al giorno
9	Piazzare la tessera corrente	700 al giorno
10	Piazzare o rimuovere un seguace su una sezione di una tessera	300 al giorno
11	Visualizzare i seguaci disponibili ad un giocatore	700 al giorno
12	Assegnare un punteggio ad un giocatore	200 al giorno
13	Aggiornare la tessera corrente	700 al giorno
14	Aggiornare il giocatore corrente	700 al giorno
15	Visualizzare le partite non concluse	5 al giorno
16	Visualizzare i giocatori di una partita	70 al giorno
17	Visualizzare le tessere piazzate in una partita	5 al giorno
18	Visualizzare le regioni in ordine di punteggio medio dei giocatori	2 al mese
19	Mostrare per ogni giocatore il numero di avversari con cui ha giocato	4 all'anno
20	Visualizzare per ogni regione l'espansione più giocata	1 al mese
21	Mostrare per ogni colore la probabilità statistica di vincere	1 al giorno

3.3 Schemi di navigazione e tabelle degli accessi

Di seguito sono presenti le tabelle che mostrano gli ingressi delle operazioni menzionate in precedenza. Inoltre, quando appropriato, sono stati inclusi gli schemi di navigazione corrispondenti. Per quanto riguarda il calcolo dei costi, si attribuisce un peso doppio agli accessi in scrittura rispetto a quelli in lettura.

Operazione 1 - Creare una nuova partita

La creazione della partita richiede una grande quantità di altre operazioni. Andremo successivamente nel dettaglio, per il momento segue la somma delle operazioni:

Operazione	Accessi
Ricerca giocatori	12S
Selezione Server	1S
Selezione Expansion	1S
Creazione Tiles	4080S + 1300L
Creare Meeple	96S
	Totale: 4190S + 1300L → 10 al giorno

Operazione 1.1 - Creazione PlayerInGame

Ad ogni partita partecipano 4 giocatori.

Concetto	Costrutto	Accessi	Tipo
Participation(Player)	R	4	S
PlayerInGame	R	4	S
Participation(Game)	R	4	S
Totale: 12S → 10 al giorno			

Operazione 1.2 - Selezione Server

Concetto	Costrutto	Accessi	Tipo
Hosting	R	1	S
Totale: 1S → 10 al giorno			

Operazione 1.3 - Selezione Expansion

In media nella creazione di una partita viene selezionata 1 espansione.

Concetto	Costrutto	Accessi	Tipo
Using	R	1	S
Totale: 1S → 10 al giorno			

Operazione 1.4 - Creazione Tiles

Concetto	Costrutto	Accessi	Tipo
Classification(TileType)	R	70	S
Tile	E	70	S
Description	R	325	L
TileTypeConfiguration	E	325	L
Composition	R	325	L
Regarding	R	325	L
Classification(GamesetType)	R	150	S
Gameset	E	150	S
Classification(TileSectionType)	R	910	S
Location	R	910	S
Subdivision	R	910	S
TileSection	E	910	S
		Totale: 4080S + 1300L → 10 al giorno	

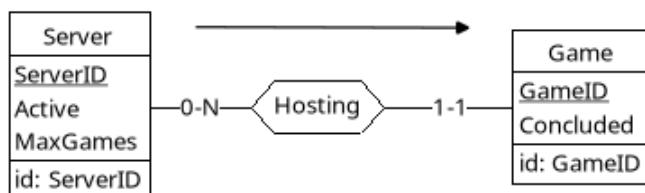
Operazione 1.5 - Creare Meeple

Ricordiamo che in una partita ogni giocatore ha 8 meeple a disposizione.

Concetto	Costrutto	Accessi	Tipo	
Classification	R	32	S	
Meeple	E	32	S	
Owning	R	32	S	
		Totale: 96S → 10 al giorno		

Operazione 2 - Visualizzare i server di gioco disponibili

Per visualizzare i server di gioco disponibili dobbiamo verificare se la quantità di partite attualmente attive su quel server sono superiori all'attributo "MaxGamesCount".



Concetto	Costrutto	Accessi	Tipo
Game	E	10	L
Hosting	R	10	L
Server	E	10	L
Totale: 30L → 10 al giorno			

Operazione 3 - Visualizzare tutte le espansioni

Concetto	Costrutto	Accessi	Tipo
Expansion	E	4	L
Totale: 4L → 10 al giorno			

Operazione 4 - Creare un nuovo giocatore

Concetto	Costrutto	Accessi	Tipo
Player	E	1	S
Totale: 1S → 40 al giorno			

Operazione 5 - Visualizzare tutti i colori

Concetto	Costrutto	Accessi	Tipo
Color	E	10	L
Totale: 10L → 40 al giorno			

Operazione 6 - Visualizzare il giocatore corrente

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	4	L
Totale: 4L → 700 al giorno			

Operazione 7 - Visualizzare la tessera corrente

Concetto	Costrutto	Accessi	Tipo
Tile	E	70	L
Totale: 70L → 700 al giorno			

Operazione 8 - Ruotare la tessera corrente

La rotazione di una tessera prevede che le rispettive sezioni, tranne la centrale, cambino il proprio tipo.

Concetto	Costrutto	Accessi	Tipo
Tile	E	1	L
Subdivision	R	12	L
TileSection	E	12	L
Classification	R	12	L
Classification	R	12	S
TileSectionType	E	12	L
Totale: 12S + 49L → 300 al giorno			

Operazione 9 - Piazzare la tessera corrente

Piazzare la tessera corrente porta all'unione con le tessere circostanti, il che implica la modifica dei gameset presenti nelle section. In media sono presenti 2 tile attorno ad una tile appena piazzata, dunque, in media vengono uniti 4 gameset per piazzamento.

Concetto	Costrutto	Accessi	Tipo
Tile	E	1	L
Tile	E	1	S
Subdivision	R	12	L
TileSection	E	12	L
Location	R	12	L
Location	R	12	S
Gameset	R	4	S
Classification(Gameset)	R	4	S
		Totale: $21S + 37L \rightarrow 700$ al giorno	

Operazione 10 - Piazzare o rimuovere un seguace su una sezione di una tessera

Concetto	Costrutto	Accessi	Tipo
Meeple	E	1	L
Meeple	E	1	S
Placement	E	1	L
Placement	E	1	S
		Totale: $2S + 2L \rightarrow 300$ al giorno	

Operazione 11 - Visualizzare i seguaci disponibili ad un giocatore

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	1	L
Owning	R	8	L
Meeple	E	8	L
		Totale: $17L \rightarrow 700$ al giorno	

Operazione 12 - Assegnare un punteggio ad un giocatore

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	1	L
PlayerInGame	E	1	S
		Totale: $1S + 1L \rightarrow 200$ al giorno	

Operazione 13 - Aggiornare la tessera corrente

Aggiornare la tessera corrente implica modificare sia la Tile successiva che la corrente.

Concetto	Costrutto	Accessi	Tipo
Tile	E	2	L
Tile	E	2	S
		Totale: 2S + 2L → 700 al giorno	

Operazione 14 - Aggiornare il giocatore corrente

Aggiornare il giocatore corrente implica modificare sia il PlayerInGame successivo che il corrente.

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	2	L
PlayerInGame	E	2	S
		Totale: 2S + 2L → 700 al giorno	

Operazione 15 - Visualizzare le partite non concluse

Concetto	Costrutto	Accessi	Tipo
Game	E	10	L
		Totale: 10L → 5 al giorno	

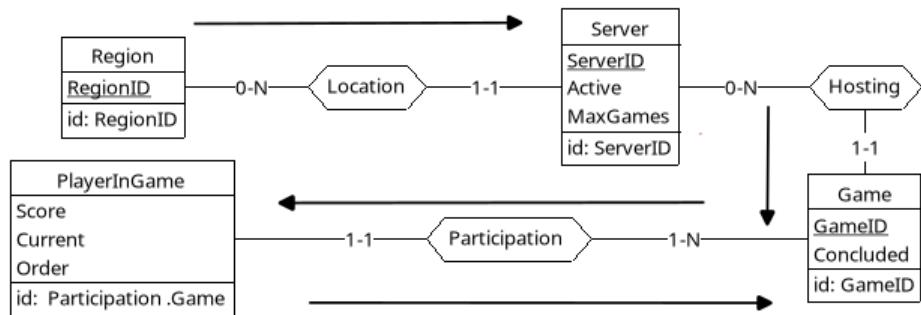
Operazione 16 - Visualizzare i giocatori di un partita

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	4	L
Participation	R	4	L
		Totale: 8L → 700 al giorno	

Operazione 17 - Visualizzare le tessere piazzate in una partita

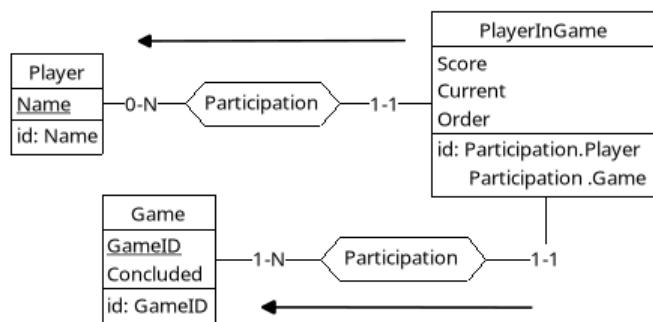
Concetto	Costrutto	Accessi	Tipo
Tile	E	70	L
Composition	R	70	L
		Totale: 140L → 5 al giorno	

Operazione 18 - Visualizzare le regioni in ordine di punteggio medio dei giocatori



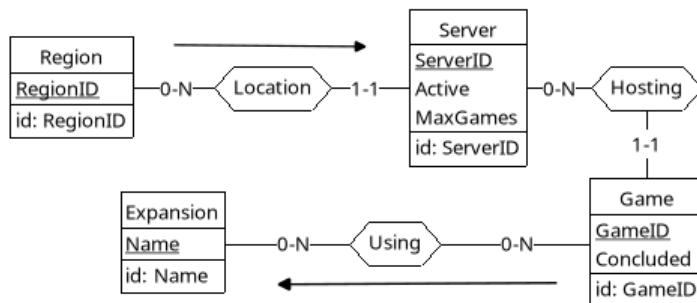
Concetto	Costrutto	Accessi	Tipo
Region	E	6	L
Location	R	10	L
Server	E	10	L
Hosting	R	10	L
Game	E	10	L
Participation	R	40	L
PlayerInGame	E	40	L
Totale: 126L → 2 al mese			

Operazione 19 - Mostrare per ogni giocatore il numero di avversari con cui ha giocato



Concetto	Costrutto	Accessi	Tipo
Player	E	40	L
Participation(Player)	R	40	L
PlayerInGame	R	40	L
Participation(Game)	R	40	L
		Totale: 160L → 4 all'anno	

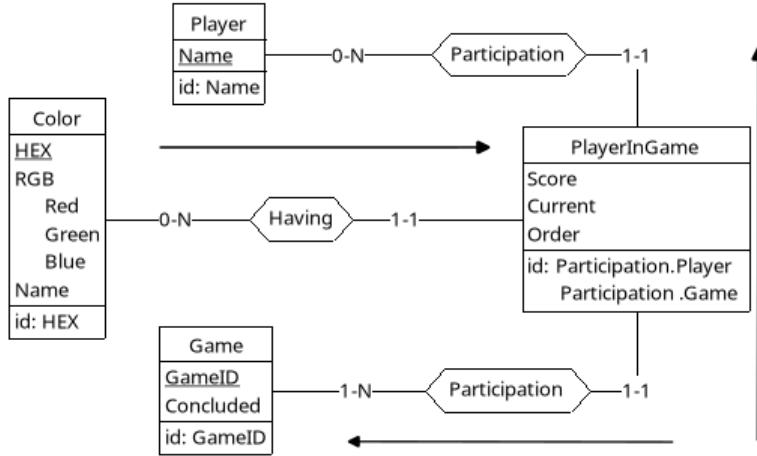
Operazione 20 - Visualizzare per ogni regione l'espansione più giocata



Concetto	Costrutto	Accessi	Tipo
Region	E	6	L
Location	R	10	L
Server	E	10	L
Hosting	R	10	L
Game	E	10	L
Using	R	10	L
Expansion	E	4	L
		Totale: 70L → 1 al mese	

Operazione 21 - Mostrare per ogni colore la probabilità statistica di vincere

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	40	L
Participation	R	40	L
Game	E	10	L
Having	R	40	L
Color	E	10	L
		Totale: 140L → 1 al giorno	



3.4 Raffinamento dello schema

Eliminazione delle gerarchie

Per l'eliminazione della gerarchia TileType si è scelto di rimuovere l'entità BasicTile e collassare verso l'alto l'entità ExpansionTile. Questo è stato possibile perché si è deciso di creare un'espansione base contenente tutti i TileType contenuti in BasicTile, rendendo quest'ultima entità inutile. La stessa strategia è stata adottata anche per le gerarchie MeepleType e GamesetType in quanto la situazione era la medesima.

Eliminazione degli attributi composti

Nello schema l'entità Tile contiene l'attributo composito Position che è stato quindi diviso nelle sue sotto-componenti. Sarà poi compito dell'applicazione accertarsi che tali attributi siano coerenti l'uno con l'altro.

Eliminazione degli identificatori esterni

Nello schema E/R sono state eliminate le seguenti relazioni:

- Participation, importando name da players in players_in_game
- Participation, importando id da games in players_in_game
- Having, importando hex da colors in players_in_game
- Owning, importando player_name e game_id da players_in_game in meeples

- Placement, importando name da continents in regions
- Specifically, importando name da cardinal_points in regions
- Location, importando id da regions in servers
- Hosting, importando id da servers in games
- Using, reificata importando id da games e name da expansions
- Composition, importando id da games in tiles
- Placement, importando id da meeples in tile_sections
- Subdivision, importando game_id e order da tiles in tile_sections
- Location, importando id da gamesets in tile_sections
- Preceding, importando name da tile_section_types in tile_section_types
- Regarding, importando name da tile_section_types in tile_types_configurations
- Composition, importando name da gameset_types in tile_types_configurations
- Description, importando name da tile_types in tile_types_configurations
- Classification, importando name da meeple_types in meeples
- Classification, importando name da tile_section_types in tile_sections
- Classification, importando name da gameset_types in gamesets
- Classification, importando name da tile_types in tiles
- Using, importando name da expansions in tile_types
- Using, importando name da expansions in gameset_types
- Using, importando name da expansions in meeple_types

In tile_section_types abbiamo scelto di importare due volte name da se stessa per avere sia la successiva che la precedente TileSection.

Scelta delle chiavi primarie

La maggior parte delle chiavi primarie sono già evidenziate in modo chiaro nello schema. Da notare l'entità Tile che è identificata tramite l'id della partita e l'ordine preciso in cui viene giocata. Per quanto riguarda l'entità TileSection questa è invece identificata tramite la Tile e il TileSectionType che è univoco all'interno di una Tile. Infine, l'entità TileTypeConfiguration è identificata con il TileType, il TileSectionType e un id aggiuntivo usato per distinguere i vari insiemi di GamesetType uguali presenti sulla Tile di quel TileType. Questo è obbligatorio in quanto su una stessa Tile potrebbero esserci due Gameset da considerare separati con stesso GamesetType.

3.5 Analisi delle ridondanze

Ridondanza riguardo al campo Placed in Meeple

Le operazioni 10 e 11 fanno uso del campo Placed in Meeple. Questo costituisce una ridondanza in quanto per verificare se un seguace è stato collocato su una sezione è sufficiente verificare se è presente l'associazione Placement. Nel caso dell'operazione 10, con ridondanza si ha:

Concetto	Costrutto	Accessi	Tipo
Meeple	E	1	L
Meeple	E	1	S
Placement	E	1	L
Placement	E	1	S
		Totale: 2S + 2L → 300 al giorno	
Costo Totale = $(2*2 + 2) * 300 = 1800$			

Mentre senza ridondanza si ha:

Concetto	Costrutto	Accessi	Tipo
Meeple	E	1	L
Placement	E	1	L
Placement	E	1	S
		Totale: 2L + 1S → 300 al giorno	
Costo Totale = $(2 + 2) * 300 = 1200$			

Potrebbe quindi sembrare che l'utilizzo della ridondanza sia svantaggioso.

Per l'operazione 11 con ridondanza si ottiene:

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	1	L
Owning	R	8	L
Meeple	E	8	L
		Totale: 17L → 700 al giorno	
Costo Totale = 17*700 = 11900			

Senza la ridondanza:

Concetto	Costrutto	Accessi	Tipo
PlayerInGame	E	1	L
Owning	R	8	L
Meeple	E	8	L
Placement	E	1	L
		Totale: 25L → 700 al giorno	
Costo Totale = 25*700 = 17500			

Si decide di prediligere l'utilizzo della ridondanza in quanto il costo totale dell'operazione 10 e 11 con ridondanza è minore rispetto a senza.

Ridondanza riguardo al campo Concluded in Game

Il campo Concluded, utilizzato nell'operazione 15, è una ridondanza in quanto per dichiarare una partita conclusa è sufficiente verificare se tutte le relative tessere sono state piazzate. Vediamo lo schema con la ridondanza:

Concetto	Costrutto	Accessi	Tipo
Game	E	1	L
		Totale: 1L → 5 al giorno	
Costo Totale = 1*5 = 5			

Senza ridondanza invece:

Concetto	Costrutto	Accessi	Tipo
Game	E	1	L
Composition	R	70	L
Tile	E	70	L
		Totale: 141L → 5 al giorno	
Costo Totale = 141*5 = 705			

3.6 Traduzione di entità e associazioni in relazioni

players(**name**)

colors(**hex**, red, green, blue, **name**)

games(**id**, server_id, concluded)
FK: server_id REFERENCES servers(id)

players_in_game(**player_name**, **game_id**, color_hex, score, current, order)
FK: player_name REFERENCES players(name)
FK: game_id REFERENCES games(id)
FK: color_hex REFERENCES colors(hex)

games_expansions(**expansion_name**, **game_id**)
FK: game_id REFERENCES games(id)
FK: expansion_name REFERENCES expansions(name)

expansions(**name**)

meeple_types(**name**, quantity, strength, expansion_name)
FK: expansion_name REFERENCES expansions(name)

meeples(**id**, owner_player_name, owner_game_id, type_name, placed)
FK: owner_player_name REFERENCES players_in_game(player_name)
FK: owner_game_id REFERENCES players_in_game(game_id)
FK: type_name REFERENCES meeple_types(name)

gamesets(**id**, type_name, points, closed)
FK: type_name REFERENCES gameset_types(name)

gameset_types(**name**, starting_points, endgame_ratio, expansion_name)
FK: expansion_name REFERENCES expansions(name)

tile_section_types(**name**, next_name*, previous_name*)
FK: next_name REFERENCES tile_section_types(name)
FK: previous_name REFERENCES tile_section_types(name)

tile_sections(**type_name**, **tile_order**, **tile_game_id**, gameset_id, meeple_id*, closed)
FK: type_name REFERENCES tile_section_types(name)
FK: tile_order REFERENCES tiles(order)
FK: tile_game_id REFERENCES tiles(game_id)
FK: meeple_id REFERENCES meeples(id)
FK: gameset_id REFERENCES gamesets(id)

tiles(**order**, **game_id**, type_name, rotation_count, x_coordinate*, y_coordinate*, current)

 FK: type_name REFERENCES tile_types(name)

 FK: game_id REFERENCES games(id)

tile_types(**name**, quantity, expansion_name)

 FK: expansion_name REFERENCES expansions(name)

tile_type_configurations(id, **tile_section_type_name**, **tile_type_name**, gameset_type_name, closed, pennant)

 FK: tile_section_type_name REFERENCES tile_section_types(name)

 FK: tile_type_name REFERENCES tile_types(name)

 FK: gameset_type_name REFERENCES gameset_types(name)

regions(**id**, continent_name, cardinal_point_name*)

 FK: continent_name REFERENCES continents(name)

 FK: cardinal_point_name REFERENCES cardinal_points(name)

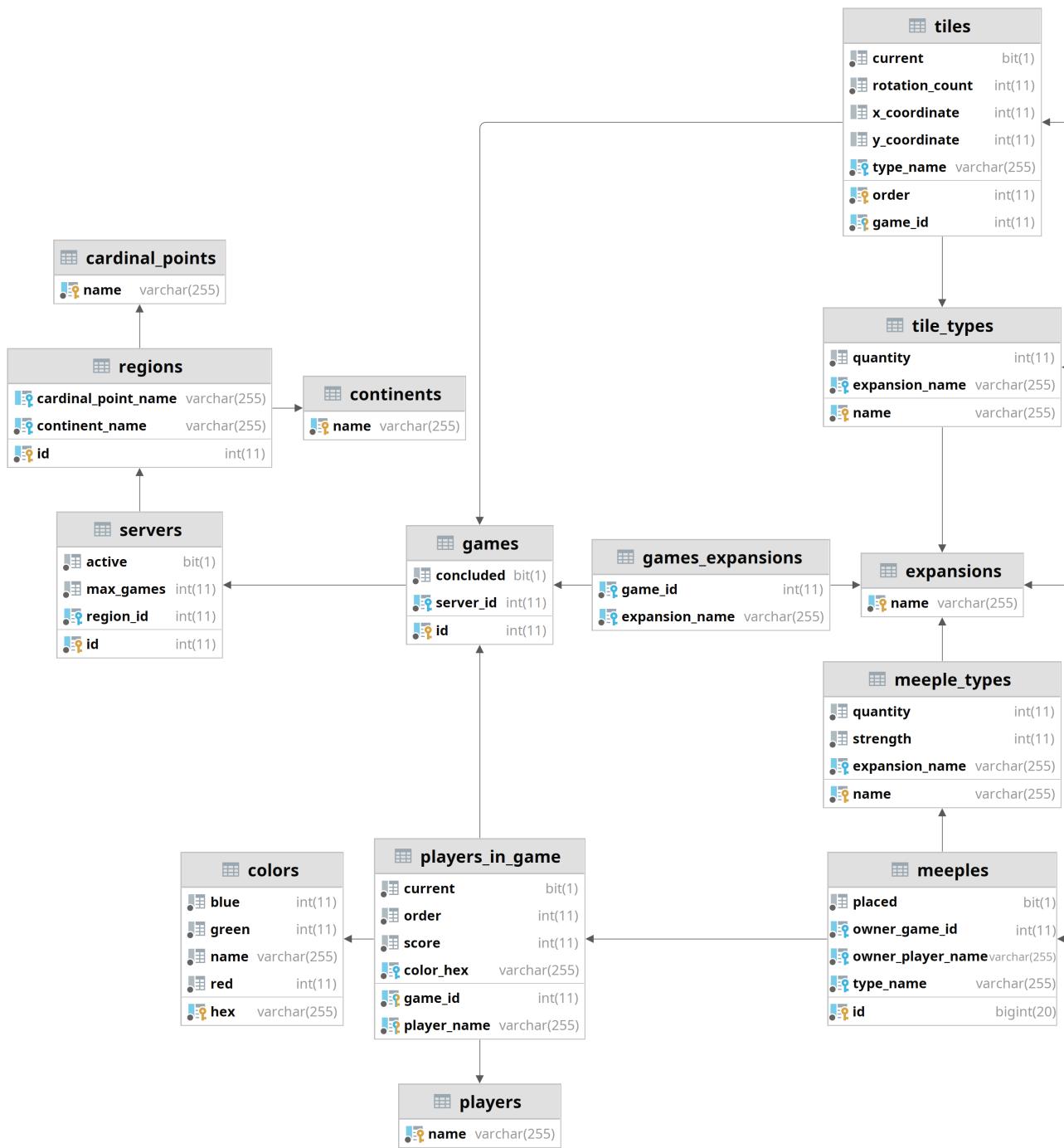
continents(**name**)

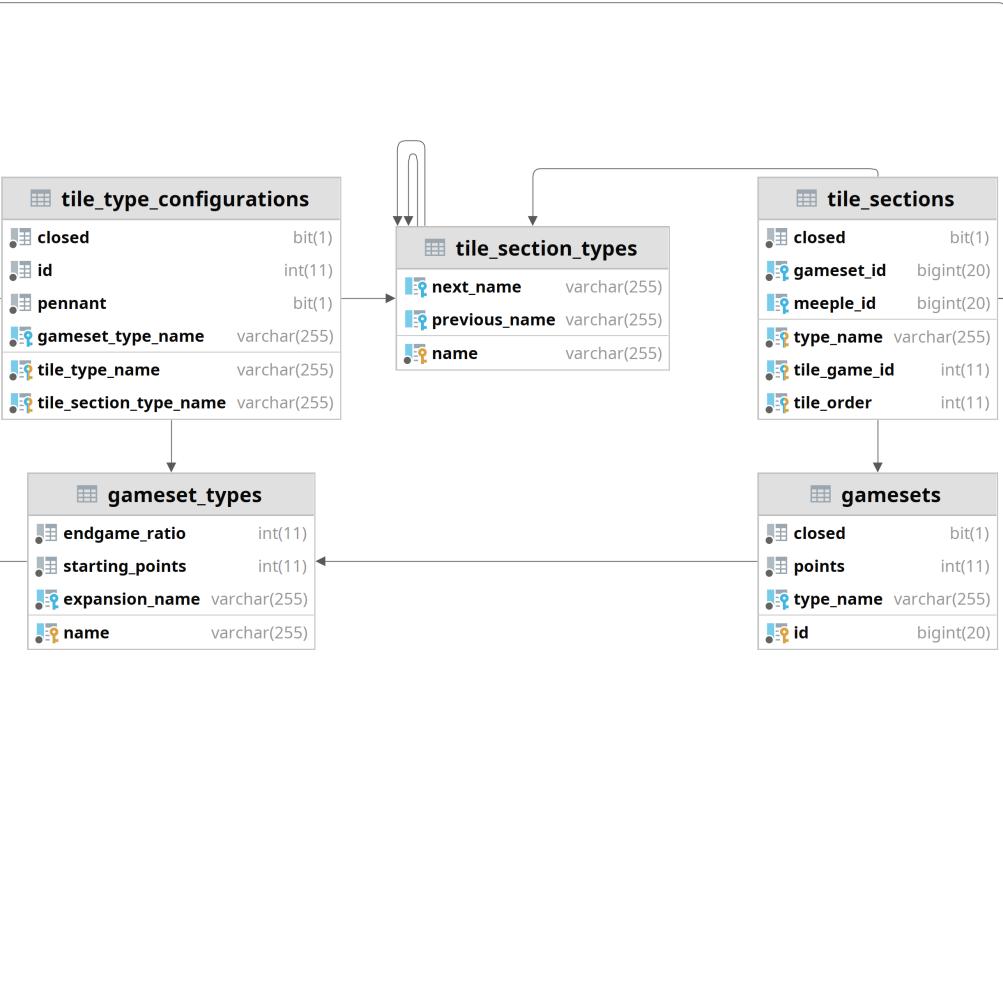
cardinal_points(**name**)

servers(**id**, region_id, active, max_games)

 FK: region_id REFERENCES regions(id)

3.7 Schema relazionale finale





3.8 Traduzione delle operazioni in query SQL

OP1 - Creare una nuova partita

Nel caso in cui i giocatori scelti non esistano nel database vengono creati tramite l'operazione 4. Viene poi creata la partita assieme ai vari players_in_game che avranno come game_id l'id della partita appena creata. Vengono poi creati i seguaci, le strutture e le tessere con le relative sezioni in accordo con le espansioni selezionate alla creazione del gioco.

```
INSERT INTO games (concluded, server_id)
VALUES (FALSE, ?);

INSERT INTO players_in_game (current, 'order', score, game_id,
    player_name, color_hex)
VALUES (FALSE, ?, 0, ?, ?, ?);

INSERT INTO meeples (placed, owner_game_id, owner_player_name,
    type_name)
VALUES (FALSE, ?, ?, ?);

INSERT INTO gamesets (closed, points, type_name)
VALUES (FALSE, 0, ?);

INSERT INTO tiles ('order', current, rotation_count,
    x_coordinate, y_coordinate, game_id, type_name)
VALUES (?, FALSE, 0, null, null, ?, ?);

INSERT INTO tile_sections (closed, type_name, tile_game_id,
    tile_order, gameset_id, meeple_id)
VALUES (FALSE, ?, ?, ?, ?, null);
```

OP2 - Visualizzare i server di gioco disponibili

Un server di gioco è disponibile solo se il numero di partite ancora in corso su quel server è inferiore al suo campo max_games.

```
SELECT *
FROM servers AS s
WHERE active=true AND max_games > (SELECT COUNT(*)
    FROM games AS g
    WHERE g.server_ID=s.ID AND g.concluded = FALSE);
```

OP3 - Visualizzare tutte le espansioni

```
SELECT *
FROM expansions;
```

OP4 - Creare un nuovo giocatore

```
INSERT INTO players (name)
VALUES (?);
```

OP5 - Visualizzare tutti i colori

```
SELECT *
FROM colors;
```

OP6 - Visualizzare il giocatore corrente

```
SELECT *
FROM players_in_game
WHERE game_id = ? AND current = TRUE;
```

OP7 - Visualizzare la tessera corrente

```
SELECT *
FROM tiles
WHERE game_id = ? AND current = TRUE;
```

OP8 - Ruotare la tessera corrente

Per ruotare la tessera corrente bisogna cambiare il tipo delle sezioni ad essa associate, essendo queste parte della chiave si è costretti a rimuoverle per poi re-inserirle aggiornate. Per ogni sezione della tessera corrente bisogna solo aggiornare l'attributo type_name con il valore dell'attributo next_name del tipo della sezione in questione, il resto rimane invariato.

```
SELECT *
FROM tile_sections
WHERE tile_game_id = ? AND tile_order = ?;
```

```

DELETE
FROM tile_sections
WHERE tile_game_id = ? AND tile_order = ?;

-- Per ogni sezione di tile selezionata con la prima query
INSERT INTO tile_sections (closed, type_name, tile_game_id,
    tile_order, gameset_id, meeple_id)
VALUES (?, ?, ?, ?, ?, ?);

```

OP9 - Piazzare la tessera corrente

```

UPDATE tiles
SET x_coordinate = ?,
y_coordinate = ?
WHERE game_id = ? AND current = TRUE;

```

OP10 - Piazzare o rimuovere un seguace su una sezione di una tessera

Nel caso della rimozione di un seguace è invece necessario impostare il relativo attributo placed a false. Inoltre, bisogna anche impostare a null l'attributo meeple_id della sezione sulla quale era piazzato.

```

UPDATE meeples
SET placed = TRUE
WHERE id = ?;

UPDATE tile_sections
SET meeple_id = ?
WHERE type_name = ? AND tile_game_id = ? AND tile_order = ?;

```

OP11 - Visualizzare i seguaci disponibili ad un giocatore

Un seguace è disponibile ad un giocatore se non è piazzato.

```

SELECT *
FROM meeples
WHERE owner_player_name = ? AND owner_game_id = ? AND placed =
FALSE;

```

OP12 - Assegnare un punteggio ad un giocatore

```
UPDATE players_in_game
SET score = ?
WHERE game_id = ? AND player_name = ?;
```

OP13 - Aggiornare la tessera corrente

Il valore della wildcard per l'attributo order della seconda query si ricava incrementando di 1 l'attributo order della tessera che era prima corrente.

```
UPDATE tiles
SET current = FALSE
WHERE game_id = ? AND current = TRUE;

UPDATE tiles
SET current = TRUE
WHERE game_id = ? AND order = ?;
```

OP14 - Aggiornare il giocatore corrente

Il valore della wildcard per l'attributo order della seconda query si ricava incrementando di 1 l'attributo order del giocatore che era prima corrente e applicandogli il modulo con il numero di giocatori della partita.

```
UPDATE players_in_game
SET current = FALSE
WHERE game_id = ? AND current = TRUE;

UPDATE players_in_game
SET current = TRUE
WHERE game_id = ? AND order = ?;
```

OP15 - Visualizzare le partite non concluse

```
SELECT *
FROM games
WHERE concluded = FALSE;
```

OP16 - Visualizzare i giocatori di una partita

```
SELECT *
FROM players_in_game
WHERE game_id = ?;
```

OP17 - Visualizzare le tessere piazzate in una partita

```
SELECT *
FROM tiles
WHERE game_id = ? AND x_coordinate IS NOT NULL AND y_coordinate
IS NOT NULL;
```

OP18 - Visualizzare le regioni in ordine di punteggio medio dei giocatori

In questo caso vengono considerate solo le partite concluse.

```
SELECT R.id, AVG(P.score) AS AveragePlayerScore
FROM ((regions R INNER JOIN servers S on R.id = S.region_id) INNER
      JOIN games G ON S.id = G.server_id) INNER JOIN players_in_game P
      ON G.id = P.game_id
WHERE G.concluded = TRUE
GROUP BY R.id
ORDER BY AveragePlayerScore DESC;
```

OP19 - Mostrare per ogni giocatore il numero di avversari con cui ha giocato

Nella subquery per ogni giocatore si conta con quanti altri giocatori diversi e distinti ha giocato.

```
SELECT P.player_name, (SELECT COUNT(DISTINCT P2.player_name) AS
                           EnemiesCount
                      FROM players_in_game P2 INNER JOIN games G2 on P2.game_id = G2.id
                      WHERE G2.id IN (SELECT DISTINCT P3.game_id
                                      FROM players_in_game P3
                                      WHERE P3.player_name = P.player_name) AND P2.player_name <>
                           P.player_name) AS EnemiesCount
FROM players_in_game P INNER JOIN games G on P.game_id = G.id
GROUP BY P.player_name
ORDER BY EnemiesCount DESC;
```

OP20 - Visualizzare per ogni regione l'espansione più giocata

Per questa operazione si è deciso di non considerare l'espansione base in quanto viene usata in ogni partita. Nella subquery per ogni regione si conta in quante partite è stata usata ogni espansione. Nella query esterna si seleziona solo l'espansione più giocata per ogni regione.

```
SELECT SQ.id, SQ.expansion_name, MAX(SQ.GamesCount) AS MaxGamesCount
FROM (SELECT R.id, GE.expansion_name, COUNT(GE.expansion_name) AS
      GamesCount
     FROM ((regions R INNER JOIN servers S ON R.id = S.region_id)
           INNER JOIN games G ON G.server_id = S.id) INNER JOIN
          games_expansions GE ON GE.game_id = G.id
      WHERE GE.expansion_name <> 'Basic'
      GROUP BY R.id, GE.expansion_name) AS SQ
GROUP BY SQ.id
ORDER BY SQ.id ASC;
```

OP21 - Mostrare per ogni colore la probabilità statistica di vincere

Viene effettuata una subquery per ricavare i vincitori delle partite e in questo modo tramite un raggruppamento e una somma viene individuato il numero di vincitori per ogni colore. Viene infine effettuata una divisione tra il numero dei vincitori e il totale delle tuple così da ottenere la probabilità di vincere sulla base del colore scelto.

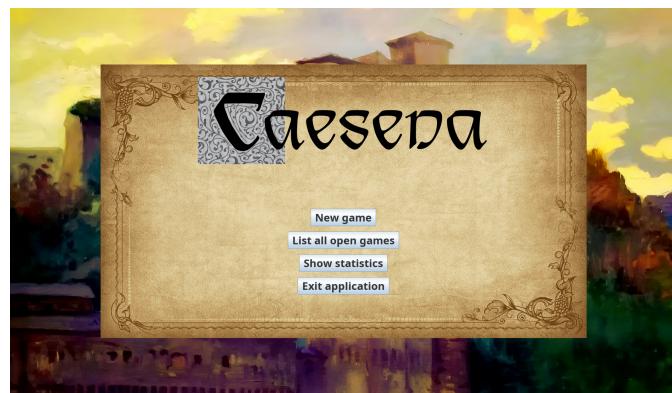
```
SELECT c.name, (SUM(CASE WHEN pig.Score = max_scores.max_score THEN
      1 ELSE 0 END) / COUNT(*)) AS WinProbability
FROM colors c JOIN players_in_game pig ON c.hex = pig.color_hex
      JOIN (SELECT game_id, MAX(Score) AS max_score
            FROM players_in_game
            GROUP BY game_id) max_scores ON pig.game_id = max_scores.game_id
      JOIN games g ON pig.game_id = g.id
WHERE g.concluded = TRUE
GROUP BY c.hex, c.name;
```

4 Progettazione dell'applicazione

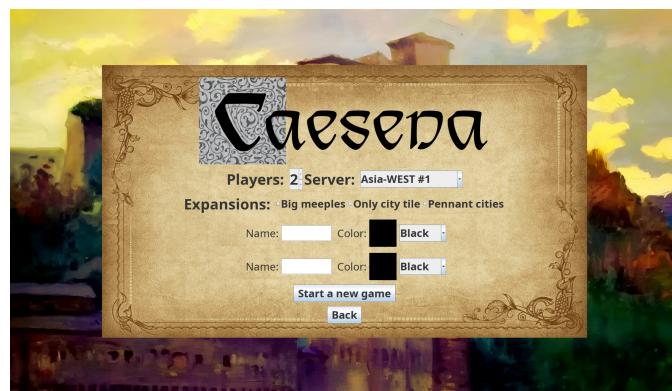
4.1 Descrizione dell'architettura

L'applicazione realizzata è stata scritta in linguaggio Java e si interfaccia con il database usando lo strumento di ORM Hibernate. Il database risiede in locale e come DBMS è stato scelto MySQL. Dal punto di vista architettonico l'applicazione implementa il pattern MVC: è quindi la parte di Controller ad occuparsi di interagire con il database.

La schermata iniziale presenta 4 pulsanti che permettono di iniziare una nuova partita, continuare una partita in corso, visualizzare statistiche relative alle partite giocate e chiudere l'applicazione.



La schermata per l'inizio di una nuova partita presenta la scelta del server in cui giocare la partita, le espansioni da usare, il numero di giocatori, e per ciascuno, dei campi in cui inserirne il nome e il relativo colore. Dopo aver selezionato e impostato a piacimento le informazioni dei giocatori, si dovrà semplicemente cliccare sul pulsante "Inizia una nuova partita".



La schermata per il proseguimento di una partita in corso presenta una casella combinata per scegliere di quale giocatore si vuole visualizzare l'elenco delle partite in corso. Apparirà quindi una tabella in cui in ogni riga saranno presenti le informazioni di una partita diversa ed un pulsante "Unisciti".

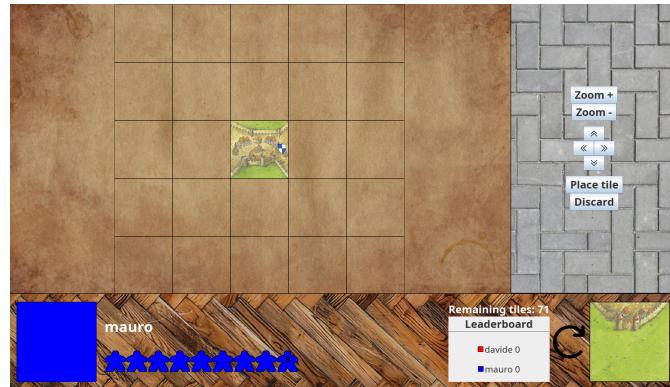


La schermata per la visualizzazione di statistiche relative alle partite giocate presenta una tabella diversa per ognuna di esse. Degli esempi di statistiche sono l'espansione non base più giocata in ogni regione e con quanti altri giocatori diversi ogni giocatore ha giocato.



Una volta avviata la partita verrà mostrata la schermata di gioco il quale contenuto principale è il tabellone che mostra le tessere piazzate con i relativi seguaci sopra. In basso a sinistra dello schermo vengono mostrati colore e nome del giocatore corrente con i suoi seguaci, mentre alla destra si trovano la classifica dei giocatori e l'immagine della tessera corrente con il pulsante per ruotarla. A destra dello schermo si presentano invece i controlli del tabellone che ne gestiscono lo Zoom e ne permettono lo spostamento al suo interno. Premendo il pulsante "Piazza tessera" questa verrà piazzata solo in caso sia stata selezionata una posizione corretta nel tabellone. Premendo invece il

pulsante "Scarta" verrà scartata la tessera solo se non piazzabile. Dopo aver piazzato la tessera apparirà la casella combinata che permette di scegliere il tipo di seguace da piazzare quando si preme il pulsante "Piazza seguace".



Alla fine della partita, verrà mostrata la classifica finale nella quale ogni giocatore vedrà il proprio punteggio conclusivo e avrà la possibilità di tornare alla schermata iniziale o di uscire dal gioco.

Il gioco supporta anche la lingua inglese nel caso il sistema operativo non utilizzi la lingua italiana, il testo dei pulsanti alla quale si riferisce questo capitolo si traduce nel seguente modo:

- "Inizia una nuova partita" → "Start a new game"
- "Unisciti" → "Join"
- "Piazza tessera" → "Place tile"
- "Scarta" → "Discard"
- "Piazza seguace" → "Place meeple"
- "Finisci turno" → "End turn"

Riferimenti bibliografici

- [1] Wikipedia. Pagina Wikipedia del gioco da tavolo Carcassonne. URL:
[https://it.wikipedia.org/wiki/Carcassonne_\(gioco\)](https://it.wikipedia.org/wiki/Carcassonne_(gioco)).