Casinò Manager

Abstract

“Casinò Manager” è un’applicazione gestionale utilizzata dalle principali case di gioco nel mondo.
Lo scopo del gestionale è quello di centralizzare e informatizzare tutti i dati riguardanti le proprie strutture gestite in modo da avere il pieno controllo del “*cash flow*” della struttura.
Tramite Casinò Manager possono essere eseguite operazioni di *Fraud Detection* e statistica generale sulle giocate.
Vengono inoltre gestite le risorse umane (turni di lavoro e anagrafica) e gli ambienti della struttura stessa (Sale e postazioni di gioco).
Inoltre viene tracciata l’intera clientela, in modo da poter fornire un servizio di consultazione delle proprie transazioni (giocate vincenti/perdenti e ricariche del credito di gioco) ai clienti.
Infine “Casinò Manager” permette di gestire i tornei svolti all’interno della struttura, gestendo partecipanti, match di gioco e vincitori.

Analisi dei requisiti

Il progetto mira a generare una base di dati, che rappresenti il gestionale di uno o più casinò.

Ogni **casinò** è caratterizzato da: il proprio indirizzo, un nome, i propri dipendenti, delle sale e dei clienti.

Ad ogni casinò è associata una lista di **clienti**, di ognuno di essi si conoscono i dati anagrafici e il saldo.

Il **personale** rappresenta i lavoratori del casinò (croupier, cassieri, sicurezza, dirigenti, baristi), ogni persona ha la propria anagrafica (nome, cognome, etc) e il ruolo svolto all’interno della struttura.

Le **sale** rappresentano i vari ambienti del casinò, esse possono essere divise in 3 tipi: sale gioco(gambling), sale ristoro e sale congressi.
Ogni sala ha un proprio nome, un numero, e viene tenuta traccia della posizione di essa all’interno della struttura.

Le sale gambling possono avere una o più **postazioni** di gioco.

Ogni **postazione** è caratterizzata da un nome, un numero e il tipo di gioco giocato in essa.

Le diverse tipologie di **gioco** vengono rappresentate attraverso il loro tipo (Blackjack, Poker, Roulette) e la loro “versione”/nome (es. Roulette Francese o Fair Roulette).
Ogni gioco ha un breve descrizione associata ed un insieme di regole.

Ad ogni giocata da parte di un cliente viene registrata una **transazione** con le seguenti informazioni: id univoco, timestamp di quando è stata eseguita.
Ogni transazione può essere di tipo: Vincita, Perdita o ricarica.
Ad ogni ricarica del conto di gioco del cliente viene generata una transazione di tipo Ricarica, mentre per le transazioni di gioco viene utilizzato Vincita e perdita. I clienti possono partecipare a dei **Tornei** ospitati nelle sale del casinò.

Di ogni torneo vengono salvati i dati riguardanti “l’evento”, quindi premio, buy-in (quota d’entrata), la data e la sala in cui sarà svolto.

Ogni torneo è composto da un insieme di **match** che rappresentano i vari “scontri” tra i giocatori. Ad ogni match verrà tenuto conto del saldo dei giocatori per decretare il vincitore finale.

Glossario dei termini

Termine	Descrizione	Collegamenti
Casinò	una struttura di gioco	Personale, Sale, Clienti
Personale	Persona che lavora all’interno della struttura. (Dirigenti, croupier, barman, sicurezza)	Casinò, Sale, Postazioni
Gambling	Stanza/ambiente nella quale sono posizionate le postazioni da gioco	Casinò, Postazioni, Personale, Tornei
Ristoro	Stanza/Ambiente dedicata al bar/ristorante	Casinò, Personale
Congressi	Stanza/Ambiente dedicata ad eventi e presentazioni	Casinò, Personale
Postazioni	Descrive le postazioni di gioco presenti all’interno di una sala, come tavoli, giochi elettronici	Gambling, Personale, Gioco, Vincita, Perdita
Gioco	Descrive il gioco giocato nella postazione	Postazioni, Tornei
Ricarica	Transazione rappresentante la ricarica del saldo di gioco	Clienti
Perdita	Transazione di gioco perdente	Postazioni, Clienti
Vincita	Transazione di gioco vincente	Postazioni, Clienti
Clienti	Rappresenta tutta la clientela, dei casinò	Casinò, Transazioni, Tornei
Torneo	Rappresenta un torneo di un gioco	Match, Gioco, Sale, Clienti
Match	Rappresenta i vari round del torneo	Clienti, Tornei
Fraud detection	Insieme di operazioni atte a individuare transazioni fraudolente da parti dei clienti e/o lavoratori della strututra	

Operazioni tipiche

Le seguenti operazioni vengono calcolate in una settimana regolare di attività.

1. Inserimento Clienti	S	200 al giorno
2. Controllo turni di lavoro	L	100 al giorno
3. Controllo saldo clienti	L	500 al giorno
4. Controllo fraud detection	L	1 al giorno
5. Inserimento turni di lavoro	S	1000 settimanali
6. Inserimento transazioni	S	3000 giornaliere

Le seguenti operazioni vengono invece distribuite durante l'anno:

1. Inserimento Sale	S	10 alla creazione
2. Inserimento Postazioni	S	10 all'anno
3. Inserimento Giochi	S	5 all'anno
4. Inserimento Personale	S	20 all'anno
5. Inserimento tornei	S	15 all'anno
6. Inserimento Match tornei	S	80 all'anno

Progettazione Concettuale

Lista entità

Se non specificato gli attributi sono di tipo NOT NULL

Casinò: una struttura adibita al gioco		
IDCasinò	integer PK	Codice univoco del casinò
Nome	varchar(40)	Nome della struttura
Indirizzo	Attributo composto	
Via	varchar(30)	
CAP	varchar(6)	
Provincia	varchar(2)	
Stato	varchar(2)	

Personale: lavoratore della struttura		
CF	varchar(16) PK	Codice univoco
Nome	varchar(20)	
Cognome	varchar(20)	
Contatto	varchar(20)	Contatto telefonico
Ruolo	varchar(15)	Ruolo ricoperto all'interno della struttura
Indirizzo	Attributo composto	
Via	varchar(40)	
CAP	varchar(6)	
Provincia	varchar(2)	
Stato	varchar(2)	

Sale: stanze della struttura		
Numero	integer PK	Numero identificativo della stanza
Nome	varchar(20)	Nome della sala
Posizione	varchar(20)	Posizione all'interno del casinò
Tipo	Tipo sale	Uso della sala

Sale si specializza in **Gambling, Ristoro** e **Congressi**

Gambling: sale dedicate al gioco

Congressi: sale dedicate ad eventi e congressi

Ristoro: sale dedicate al consumo di alimenti

Postazioni: postazioni di gioco		
Numero	Integer PK	Numero identificativo della postazione
Nome	varchar(15)	Nome della postazione

Gioco: gioco praticato nella postazione		
Tipo	varchar(10) PK	Tipologia di gioco
Nome	varchar(10) PK	Nome della versione del gioco
Descrizione	varchar(50)	Breve descrizione del gioco

Regole	varchar(100)	Regole di gioco

Transazioni: vincita o perdita al gioco		
IDTx	integer PK	Numero transazione
Timestamp	timestamp	Tempo di esecuzione della tx
Importo	integer	

Transazioni si divide in **Vincita, Perdita** e **Ricarica**

Vincita: transazione rappresentante una vittoria ad un gioco

Perdita: transazione rappresentante una perdita ad un gioco

Ricarica: transazione rappresentante una ricarica del saldo di gioco

Clienti: clienti della struttura		
CF	varchar(16) PK	Codice univoco che rappresenta la persona
Nome	varchar(20)	
Cognome	varchar(20)	
Contatto	varchar(20)	
Saldo	money	Saldo
Indirizzo	Attributo composto	
Via	varchar(20)	
CAP	varchar(6)	
Provincia	varchar(2)	
Stato	varchar(2)	

Tornei: evento in cui più giocatori si contendono un premio		
IDTorneo	Integer PK	Codice univoco che rappresenta il torneo
Nome	varchar(20)	Nome del torneo
DataTorneo	DATE	
BuyIN	integer	Quota d'entrata
Premio	integer	Premio vincitore

Match: mano di un torneo		
IDMatch	Integer PK	Codice univoco che rappresenta il match
IsFinal	boolean	Indica l'ultimo match del torneo
Ora	TIME	Ora in cui viene svolto

Lista relazioni

Casinò-Sale: **HA**

- Un casinò può avere una o più sale (1, N)
- Una sala può avere un solo casinò (1,1)

Casinò-Personale: **DIPENDE**

- Un casinò può avere più dipendenti (1, N)
- Un dipendente può lavorare ad un solo casinò (1,1)

Casinò-Clienti: **SPENDE**

- Un casinò può avere 0 o più clienti (0, N)
- Un cliente può essere associato a più casinò (1, N)

Personale-Postazioni: **LAVORA**

- Un dipendente può lavorare a più postazioni non contemporaneamente (1, N)
- Una postazione può avere un solo lavoratore in un determinato momento (1,1)

Personale-Sale: **LAVORA SALE**

- Un dipendente può essere associato a più sale (1, N)
- Una sala può avere più dipendenti assegnati (1, N)

Sale-Postazioni: **GIOCA**

- Una sala può avere più postazioni, o nessuna in caso di sale adibite ad altri scopi (0, N)
- Una postazione può avere una sola sala assegnata (1,1)

Sale-Tornei: **SVOLTO**

- Una sala può avere un solo torneo contemporaneamente (1, 1)
- Un torneo può essere eseguito in una sola sala (1,1)

Postazioni-Transazioni: **TRANSA**

- Una postazione può essere associata a 0 o più transazioni (0, N)
- Una transazione può avere una sola postazione associata (1,1)

Postazioni-Gioco: **CARATERIZZATO**

- Una postazione può avere un solo gioco assegnato (1, 1)
- Un gioco può essere associato a nessuna postazione o più (0, N)

Gioco-Tornei: **TIPO DI GIOCO**

- Un gioco può avere nessuno o più tornei (0, N)
- Un torneo può avere un solo tipo di gioco (1,1)

Tornei-Match: **CLASSIFICA**

- Un torneo può avere uno o più match (1, N)
- Un match può avere un solo torneo associato (1,1)

Match-Clienti: **DISPUTANO**

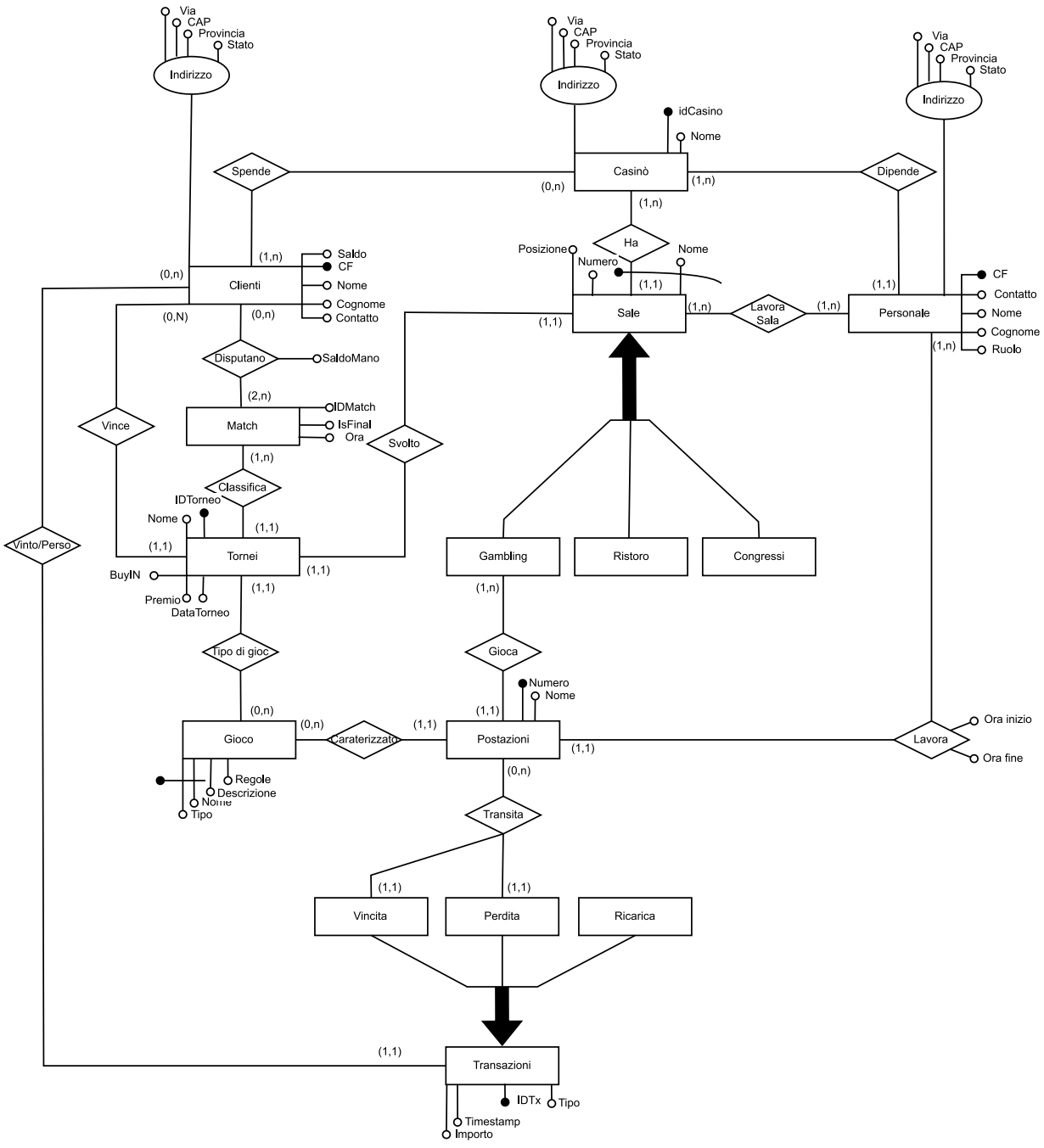
- Un match deve avere 2 o più clienti (2, N)
- Un cliente può avere uno o più match (0,1)

Clienti-Transazioni: **VINTO/PERSO**

- Un cliente può avere nessuna o più transazioni (0, N)
- Una transazione può avere un solo cliente associato (1,1)

Clienti-Tornei: **VINCE**

- Un cliente può vincere nessuno o più tornei (0, N)
- Un torneo può avere solo un vincitore (1,1)



Progettazione Logica

L'attributo "Ruolo" (di personale) e "Tipo" (di transazione) vengono trasformati in tipi custom.
L'attributo "Regole" (di gioco) viene trasformato in una nuova entità e collegato all'entità tramite una relazione molti a molti.

Analisi delle ridondanze

L'attributo "Saldo" (di cliente) potrebbe essere considerata una ridondanza in quanto potremmo calcolare facilmente il saldo di ogni cliente tramite le transazioni associate al cliente, ma questo peserebbe molto sulla base di dati:

- *Inserimento transazioni. 3000 al giorno*
- *Controllo saldo clienti. 300 al giorno.*

Ipotizzando un cliente con diversi anni di gioco alle spalle, potremmo dover accedere a migliaia di transazioni per calcolare il saldo, andando a pesare fortemente sulle tabelle interessate.

Tavola dei volumi		
Concetto	Costrutto	Volume
Vinto/Perso	R	2000000
Transazione	E	2000000
Clienti	E	1000

PRESENZA RIDONDANZA

Operazione 1:

Un cliente avrà quindi in media 2M/1000 = 2000 transazioni associate a sé.

Concetto	Costrutto	Accessi	Tipo	
Vinto/Perso	Relazione	1	S	x 3000
Vinto/Perso	Relazione	2000	L	x 3000
Clienti	Entità	1	S	x 3000

Operazione 2:

Concetto	Costrutto	Accessi	Tipo	
Clienti	Entità	1	L	x 300

Dato che gli accessi in scrittura alla base di dati costano il doppio di quelli in lettura possiamo determinare un costo giornaliero di:
3000*2+2000*3000+3000+300 = 6009300

ASSENZA RIDONDANZA

Operazione 1:

Concetto	Costrutto	Accessi	Tipo	
Vinto/Perso	Relazione	1	S	x 3000

Operazione 2:

Concetto	Costrutto	Accessi	Tipo	
Clienti	Entità	1	L	x 300
Vinto/Perso	Relazione	3000	L	x 3000

3000*2+300+3000*3000=9006300
Possiamo quindi vedere che mantenere la ridondanza è più vantaggioso.

La relazione "Vince" (tra Tornei e Clienti), può essere rimossa in quanto il cliente vincitore del torneo può essere facilmente recuperato controllando l'utente con saldo diverso da 0 nell'ultimo match del torneo.

Andando quindi a risparmiare un campo nella tabella tornei al costo di un accesso in più alla tabella Match.

Eliminazione delle generalizzazioni:

Per entrambe le generalizzazioni è stato scelto di accorpare le tre entità figlie nella genitrice.
In questo modo potremmo risparmiare tabelle e i rispettivi identificatori nelle tabelle in cui dovrebbero essere referenziate, andando ad inserire un attributo per distinguere i 3 tipi.

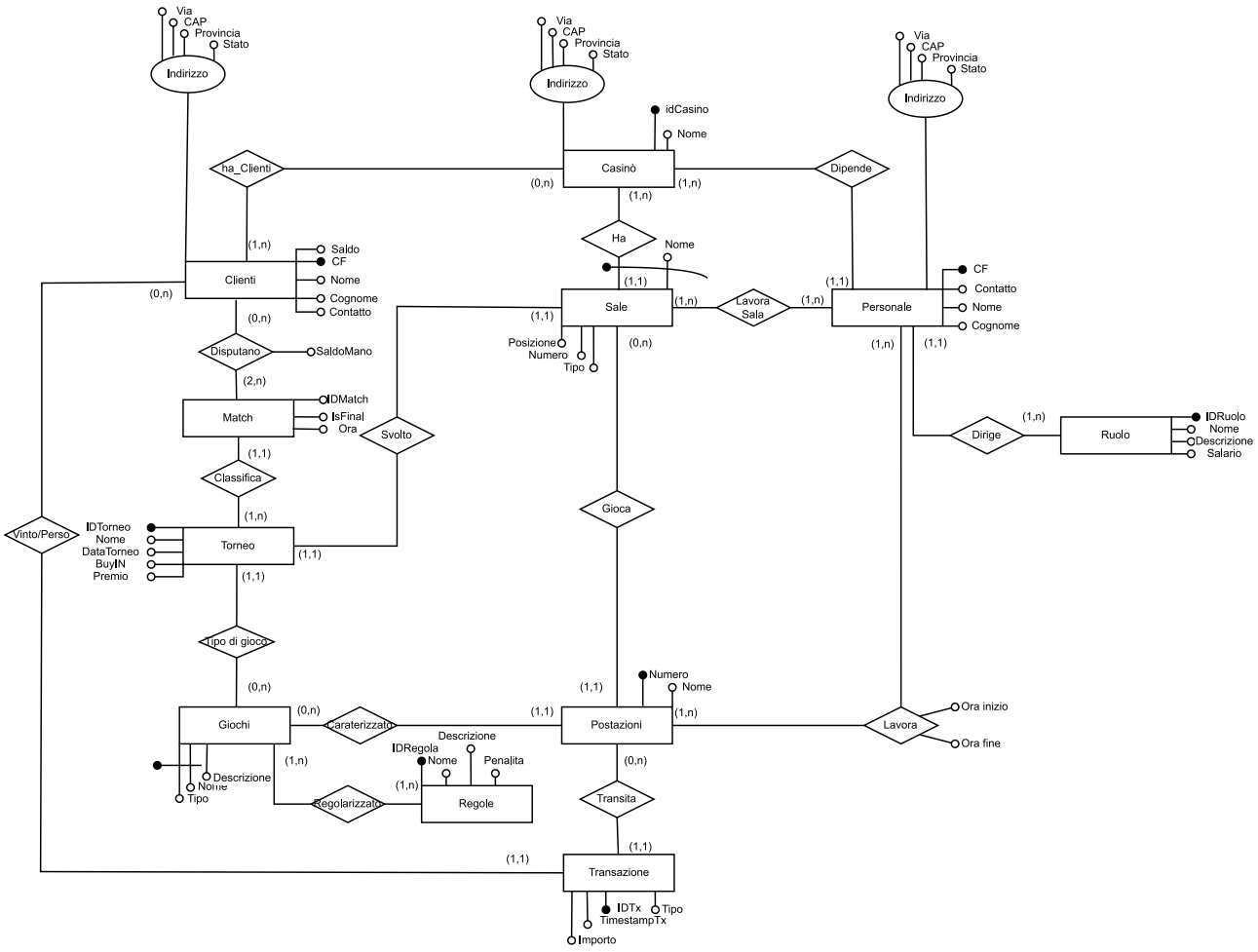
Scelta degli identificatori primari:

Per le tabelle “Clienti” e “Personale” è stato scelto di utilizzare un id univoco di tipo integer, nonostante avessimo già il campo “CF” univoco per ogni occorrenza delle tabelle.
Se avremo utilizzato il campo “CF” come chiave primaria, ci saremo ritrovati con il dato replicato in ogni tabella che lo utilizzava come chiave esterna, replicando dunque 16 Char invece di un semplice intero.

Per la tabella “Sale” è stato scelto di utilizzare una chiave primaria composta dal numero di sala e l’id del casinò a cui essa appartiene.
Il numero sala non sarebbe bastato per rendere univoca l’entità assieme alle sale degli altri casinò; perciò, si è deciso di inserire anche l’IDCasinò.

Per l’entità “Gioco” si è scelto di inserire un Id integer univoco, per facilitare la referenziazione di essa nelle tabelle in cui è impiegata.

Diagramma E-R ristrutturato:



Schema relazionale e vincoli di integrità referenziale

Gli attributi con l'asterisco possono avere valori nulli, in seguito ad ogni relazione sono riportate le chiavi esterne e i relativi attributi coinvolti

Casinò (IDCasinò, Nome, Via, CAP, Provincia, Stato)

Ruoli (IDRuolo, Nome, Descrizione, Salario)

Personale (IDPersonale, CF, Nome, Cognome, Contatto, Via, CAP, Provincia, Stato, IDRuolo, IDCasinò)

- Personale.IDRuolo -> Ruoli.IDRuolo
- Personale.IDCasinò -> Casinò.IDCasinò

Sale (Numero, Nome, Posizione, Tipo, IDCasinò)

- Sale.IDCasinò -> Casinò.IDCasinò

Dirige (IDSala, IDPersonale)

- Dirige.IDSala -> Sale.Numero
- Dirige.IDPersonale -> Personale.IDPersonale

Giochi (IDGioco, Tipo, Nome, Descrizione)

Postazioni (Numero, Nome, NomeSala, NumeroSala, IDGioco)

- Postazioni.NomeSala -> Sale.Nome
  Postazioni.NumeroSala -> Sale.Numero
- Postazioni.IDGioco -> Giochi.IDGioco

Lavora (NumeroPs, IDPersonale, Ora_Inizio, Ora_Fine)

- Lavora.NumeroPS -> Postazioni.Numero
- Lavora.IDPersonale -> Personale.IDPersonale

Clienti (IDCliente, CF, Nome, Cognome, Contatto, Via, CAP, Provincia, Stato, Saldo)

ha_clienti (IDCliente, IDCasinò)

- Ha_clienti.IDCliente -> Clienti.IDCliente)
- Ha_clienti.IDCasinò -> Clienti.IDCasinò)

Tornei (IDTorneo, Nome, DataTorneo, BuyIN, Premio, NomeSala, NumeroSala, IDGioco)

- Tornei.NomeSala -> Sale.Nome
- Tornei.NumeroSala -> Sale.Numero
- Tornei.IDGioco -> Giochi.IDGioco

Match (IDMatch, IsFinal, Ora, IDTorneo)

- Match.IDTorneo -> Tornei.IDTorneo

Disputano (IDMatch, IDCliente, SaldoMano)

- Disputano.IDMatch -> Match.IDMatch
- Disputano.IDCliente -> Clienti.IDCliente

Regole (IDRegola, Nome, Descrizione, Penalità)

Regolamento (IDGioco, IDRegola)

- Regolamento.IDGioco -> Giochi.IDGioco
- Regolamento.IDRegola -> Regole.IDRegola

Transazioni (IDTx, Tipo, TimestampTX, Importo, IDPostazione, IDCliente)

- Transazioni.IDPostazione -> Postazioni.Numero
- Transazioni.IDCliente -> Clienti.IDCliente

Query e Indice

1. Mostrare i giochi con un payout superiore alla media
   (Ex. Texas Hold'em paga più della media di tutti gli altri giochi)

```
SELECT g.Nome as Nome_Gioco, AVG(tx.Importo) as Media_Vittorie FROM Transazioni tx
INNER JOIN Postazioni AS ps ON tx.IDPostazione = ps.Numero
INNER JOIN Giochi AS g ON g.IDGioco = ps.IDGioco
WHERE tx.tipo='Vincita'
GROUP BY g.Nome
HAVING AVG(tx.Importo) > (SELECT AVG(tx.Importo) FROM Transazioni tx WHERE tx.tipo='Vincita')
```

```
+----------------+--------------------+
| nome_gioco     | media_vittorie     |
+----------------+--------------------+
| Texas hold 'em | 94413.444444444444 |
| SL CrazyOne    | 95882.602040816327 |
+----------------+--------------------+
```

2. Mostrare le statistiche dei tavoli di un casinò
   (Numero transazioni vincenti, media vincite, numero transazioni perdenti, media perdite)

```
SELECT vin.Nome,Vincite,vin.media,Perdite,per.media FROM
(SELECT ps.Nome, COUNT(IDTx) AS Vincite, AVG(Importo) AS media FROM Postazioni ps LEFT JOIN Transazioni tx ON Numero=IDPostazione
INNER JOIN Sale ON  NomeSala = Sale.Nome AND NumeroSala = Sale.Numero
WHERE tx.Tipo='Vincita' AND IDCasinò=2 GROUP BY ps.Nome,tx.Tipo) AS vin
INNER JOIN
(SELECT ps.Nome, COUNT(IDTx) AS Perdite, AVG(Importo) AS media FROM Postazioni ps LEFT JOIN Transazioni tx ON Numero=IDPostazione
INNER JOIN Sale ON  NomeSala = Sale.Nome AND NumeroSala = Sale.Numero
WHERE tx.Tipo='Perdita' AND IDCasinò=2 GROUP BY ps.Nome,tx.Tipo) AS per ON vin.Nome = per.Nome
```

```
+------+---------+----------+---------+----------+
| nome | vincite | round    | perdite | round    |
+------+---------+----------+---------+----------+
| AR1  | 8       | 10544.00 | 10      | 7416.70  |
| AR2  | 15      | 10090.80 | 6       | 11215.67 |
| FR1  | 9       | 8775.11  | 5       | 15042.80 |
| FR2  | 10      | 9436.40  | 8       | 9688.63  |
| PK1  | 10      | 9813.00  | 11      | 7557.09  |
| PK2  | 9       | 8598.89  | 5       | 7649.20  |
+------+---------+----------+---------+----------+
```

3. Mostrare il cliente che ha vinto la maggior somma di denaro ad un tipo di gioco
   (In questo caso alla “Roulette”)

```
SELECT DISTINCT Nome, Cognome, Importo FROM Clienti
INNER JOIN Transazioni ON Clienti.IDCliente=Transazioni.IDCliente
WHERE Importo = (SELECT MAX(Importo) FROM Transazioni
WHERE Tipo='Vincita' AND IDPostazione IN
(SELECT Numero FROM Postazioni WHERE IDGioco IN (SELECT IDGioco FROM Giochi WHERE Tipo ='Roulette')))
```

```
+-------+----------+---------+
| nome  | cognome  | importo |
+-------+----------+---------+
| Cobby | Wherrett | 239166  |
+-------+----------+---------+
```

4. Mostrare il vincitore di un torneo
   (In questo caso “Italian Poker Tour”)

```
SELECT c.Nome,c.Cognome, disp.SaldoMano FROM Clienti c
INNER JOIN Disputano AS disp ON disp.IDCliente = c.IDCliente
INNER JOIN Match AS m ON disp.IDMatch = m.IDMatch
WHERE m.IsFinal = true AND m.IDTorneo = 1 AND disp.SaldoMano != 0
```

```
+------+---------+-----------+
| nome | cognome | saldomano |
+------+---------+-----------+
| Mara | Breslau | 12000     |
+------+---------+-----------+
```

5. Vedere i turni di lavoro di un dipendete

```
SELECT ps.nome,ps.cognome,pst.Nome,pst.NomeSala,lv.Ora_Inizio,lv.Ora_Fine
FROM Lavora lv
INNER JOIN Personale ps ON lv.IDPersonale=ps.IDPersonale
INNER JOIN Postazioni pst ON lv.NumeroPs=pst.Numero
WHERE ps.IDPersonale=1
```

```
+--------+----------+------+-----------+---------------------+---------------------+
| nome   | cognome  | nome | nomesala  | ora_inizio          | ora_fine            |
+--------+----------+------+-----------+---------------------+---------------------+
| Erinna | Lauthian | FR1  | Sala Verde | 2020-07-01 14:00:00 | 2022-07-01 18:00:00 |
| Erinna | Lauthian | FR1  | Sala Verde | 2020-06-01 18:00:00 | 2022-06-01 22:00:00 |
| Erinna | Lauthian | FR1  | Sala Verde | 2020-05-01 10:00:00 | 2022-05-01 14:00:00 |
+--------+----------+------+-----------+---------------------+---------------------+
```

6. Dipendenti con media vittorie sopra la media durante il loro turno

```
SELECT DISTINCT p.nome,p.cognome,p.IDPersonale,alldata.mediaImporto FROM (SELECT pe.IDPersonale,AVG(tr.Importo) as mediaImporto
                FROM ((Personale as pe INNER JOIN Lavora as l ON pe.IDPersonale=l.IDPersonale)
                INNER JOIN Postazioni AS po ON l.NumeroPs=po.Numero)
                INNER JOIN Transazioni AS tr ON po.Numero=tr.IDPostazione
WHERE tr.Tipo='Perdita'
GROUP BY pe.IDPersonale
HAVING AVG(tr.importo) > (SELECT AVG(tr.importo) FROM
                ((Personale as pe INNER JOIN Lavora as l ON pe.IDPersonale=l.IDPersonale)
                INNER JOIN Postazioni AS po ON l.NumeroPs=po.Numero)
                INNER JOIN Transazioni AS tr ON po.Numero=tr.IDPostazione
                WHERE tr.Tipo='Perdita')) as alldata
                INNER JOIN Personale as p on alldata.IDPersonale=p.IDPersonale
                order by p.IDPersonale
```

nome	cognome	idpersonale	round
Erinna	Lauthian	1	136391.85
Tris	Scottini	10	136391.85
Haskel	Newick	17	136391.85
Ofelia	Bruford	23	127461.88
Melessa	Love	30	127461.88
Raquel	Pigrome	31	127461.88
Rena	Life	51	110480.69
Tarrah	Harman	52	110480.69
Lock	Wahlberg	53	110480.69
Washington	Ochterlony	60	130119.81
Augie	Campsall	61	130119.81
Ellswerth	McGuff	63	130119.81
Jami	Rohloff	64	98274.55
Gilberto	Lingard	80	98274.55
Madelene	Boykett	83	98274.55
Stefa	Overlow	86	118713.31
Mariette	Sallis	91	118713.31
Ervin	Varran	96	118713.31

Utile per identificare dipendenti distratti o malintenzionati che arrecano danni al casinò, “regalando denaro”.

Indice:

Disponendo di un grande database frequentemente interrogato si prende in considerazione l’idea di creare un index.
Nonostante l’utilizzo di indici può andare ad influire sulle prestazioni in scrittura, si è scelto comunque di implementare un indice sulla tabella transazioni.
Le transizioni effettuate possono essere lette milioni di volte in una giornata dalle app dei clienti, dalle query per l’analisi statistica, dai cartelloni pubblicitari che mostrano le vincite, dai terminali dello staff e dalla zona di ritiro contante.

Per velocizzare quest’ultime operazioni è stato generato un indice sui campi **IDPostazione** e **IDCliente**.

L’indice porterà beneficio alle operazioni:

- Controllo saldo clienti
- Controllo fraud detection

```
CREATE INDEX idx_transazione ON transazione(idpostazione,idcliente)
```

Codice C++

Documentazione del codice

Il programma realizzato nel linguaggio C++ permette di eseguire una dimostrazione di alcune interrogazioni di demo al database tramite il terminale di sistema.
È composto da un unico file .cpp da compilare con il seguente comando:

g++ query.cpp -L dependencies\lib -lpq -o CasinòManager

Il programma dipende dalle seguenti librerie necessarie al suo corretto funzionamento:

libpq.dll e libpq.lib

Il programma può essere eseguito richiamando il nome del file compilato da terminale:

./CasinòManager

All’interno del file .cpp devono venir specificati i corretti parametri di connessione al database nei campi:

PG_HOST, PG_USER, PG_PASS, PG_PORT, PG_DB

Eseguito il programma, l’utente potrà scegliere di vedere il risultato di 6 differenti query inserendo il numero della query desiderata.
Tra queste query sono presenti delle interrogazioni parametriche, ossia richiedono l’inserimento di alcuni parametri per personalizzare la query a piacimento.
All’utente verrà mostrata la lista dei parametri possibili, per facilitare l’uso dell’applicativo.

Funzioni utilizzate dal codice:

1. *PGconn* connect()*

Dati i parametri indicati nel file .cpp prova la connessione al db, in caso dovessero esserci errori stampa un messaggio d’errore.

2. *PGresult* paramExec(PGconn* conn, string queries[], int input)*

Funzione utilizzata per l’esecuzione di query parametriche, all’interno di essa le query vengono “preparate” con il parametro scelto e poi eseguite.

3. *void checkResults(PGresult* res, const PGconn* conn)*

Verifica che i risultati della query siano validi e consistenti.

4. *void printQuery(PGresult* res)*

Stampa i risultati della query al database

5. *void printParams(PGresult* res, PGconn* conn, string queries[], int input)*

Simile a *printQuery(),* viene utilizzata all’interno di *paramExec()*, per stampare i parametri da poter inserire nella query.