



Carnevale Di Acireale

Progetto Base di dati

Giuseppe Condorelli

Matricola: 1000001910

Anno 2021/2022

Università di Catania

Dipartimento di Matematica e Informatica

Corso di Informatica L-31

Indice

1. Presentazione
 - 1.1. Obiettivo
2. Descrizione dettagliata
3. Progettazione Concettuale
 - 3.1. Specifiche sui dati
 - 3.2. Operazioni sui dati
 - 3.3. Analisi dei Requisiti
 - 3.4. Schema scheletro
 - 3.5. Schema intermedio (aggiunta cardinalità associazioni)
 - 3.6. Schema finale (Aggiunti tutti gli attributi)
 - 3.7. Vincoli non esprimibili dal modello E/R
 - 3.8. Dizionario dei dati (Entità)
 - 3.9. Dizionario dei dati (Relazioni)
4. Progettazione logica
 - 4.1. Tavola dei volumi
 - 4.2. Tavola delle Frequenze
 - 4.3. Schemi delle Operazioni
 - 4.3.1. Op.1 - Aggiunta di un nuovo account
 - 4.3.2. Op.2 - Aggiunta di una nuova persona
 - 4.3.3. Op.3 - Cambio di Area di una persona
 - 4.3.4. Op.4 - Prenotazione ad uno spettacolo
 - 4.3.5. Op.5 - Cancellazione prenotazione
 - 4.3.6. Op.6 - Votazione carro
 - 4.3.7. Op.7 - Importare denaro nell'account
 - 4.3.8. Op.8 - Acquisto biglietto
 - 4.3.9. Op.9 - Visualizzare il numero di persone in circolazione in un'area
 - 4.3.10. Op.10 - Visualizzare i dettagli di un carro
 - 4.3.11. Op.11 - Visualizzare il guadagno totale

- 4.3.12. [Op.12 - Visualizzare il numero di prenotati agli spettacoli](#)
- 4.4. [Valutazione costo ridondanza attributo "N_Persone" in Area](#)
 - 4.4.1. [Valutazione costi con la ridondanza](#)
 - 4.4.2. [Valutazione costi senza ridondanza](#)
- 4.5. [Traduzioni delle associazioni molti a molti](#)
- 4.6. [Traduzioni delle associazioni uno a molti](#)
- 4.7. [Schema logico](#)
- 4.8. [Schema UML](#)
- 5. [Progettazione Fisica](#)
 - 5.1. [Implementazione fisica della Base di Dati](#)
 - 5.2. [Implementazione delle operazioni](#)
 - 5.2.1. [Op.1 - Aggiunta di un nuovo Account](#)
 - 5.2.2. [Op.2 - Aggiunta di una nuova persona](#)
 - 5.2.3. [Op.3 - Cambio di Area di una Persona](#)
 - 5.2.4. [Op.4 - Prenotazione ad uno Spettacolo](#)
 - 5.2.5. [Op.5 - Cancellazione prenotazione](#)
 - 5.2.6. [Op.6 - Votazione Carro](#)
 - 5.2.7. [Op.7 - Importare denaro nell'Account](#)
 - 5.2.8. [Op.8 - Acquisto biglietto](#)
 - 5.2.9. [Op.9 - Visualizzare il numero di persone in circolazione in un'area](#)
 - 5.2.10. [Op.10 - Visualizzare informazioni sui Carri](#)
 - 5.2.11. [Op.11 - Visualizzare l'incasso totale](#)
 - 5.2.12. [Op.12 - Visualizzare il numero di prenotati agli spettacoli](#)
 - 5.3. [Implementazione dei Trigger](#)
 - 5.3.1. [Trigger n°1 - Entrata in un'area](#)
 - 5.3.2. [Trigger n°2 - Nuova persona](#)
 - 5.3.3. [Trigger n°3 - Acquisto del Biglietto](#)
 - 5.3.4. [Trigger n°4 - Controllo prenotazione a spettacolo](#)
 - 5.3.5. [Trigger n°5 - "Prenotazione - Controllo biglietto"](#)

Presentazione

Il **Carnevale di Acireale** è un importante e famoso evento, tenutosi nella **Città di Acireale** in provincia di **Catania**, che propone svariate attività di intrattenimento, tra cui l'esposizione di magnifici ed imponenti carri che riescono sempre a catturare lo stupore di moltissime persone.

Per tale motivo esso è un'importante **fonte di vanto** nonché **principale risorsa di turismo** per la città di Acireale che, a causa dei recenti avvenimenti Covid, non sta riuscendo più a far ripartire l'iniziativa per i problemi di **contagio** e contenimento degli **assembramenti**.

Obiettivo

L'obiettivo del documento è quello di fornire la **progettazione di una Base di Dati** adatta al miglioramento dell'intero evento, volta a garantire una migliore ed efficiente **gestione del sistema di biglietteria e di intrattenimento** del Carnevale di Acireale, sperando che possa facilitare la città nell'evitare assembramenti nei luoghi di maggior concentrazione, quali **bancarelle, biglietterie, spettacoli a Piazza Duomo ed esposizione dei carri di Carnevale** (attrazione principale dell'intero evento).

Descrizione dettagliata

L'evento è organizzato seguendo un **programma** dettagliato che fornirà ai **visitatori** una visione completa di come si svolgeranno le giornate di Carnevale.

La città, durante l'evento, avrà al suo interno un **circuito**, che sarà ulteriormente diviso in diverse macro e micro **Aree**, nelle quali ci saranno diverse **attrazioni** e verranno esibiti i vari **carri di Carnevale**.

Le **persone** per poter accedere all'interno del **circuito** dovranno essere dotate di **ticket**, che potranno **acquistare** principalmente online o in un tabaccaio abilitato alla vendita di quest'ultimi.

Inoltre sarà consentito, in via del tutto eccezionale, l'**acquisto** di quest'ultimi anche in alcune bancarelle abilitate situate in tutti gli ingressi al **circuito**. Esse, però, a differenza delle edizioni passate, avranno a disposizione per i **clienti** un numero ridotto di **biglietti**, questo per incentivare ancor più le **persone** all'acquisto online ed evitare possibili assembramenti intorno ai punti di acquisto.

Le diverse **Aree** predisposte all'interno del **circuito** avranno **capienza** limitata, che verrà **tracciata** permettendo alle persone di passare il **codice QR** del proprio **biglietto** in scanner presenti in tutti gli ingressi. In questo modo sarà possibile non permettere mai il superamento del limite predisposto evitando che la folla possa concentrarsi tutta in una sola **area**.

I **biglietti**, oltre a garantire l'ingresso all'interno del circuito, daranno anche la possibilità di poter partecipare ad un solo spettacolo proposto, al quale il **partecipante** potrà fare



prenotazione utilizzando il **codice** del proprio **biglietto** direttamente dalla piattaforma.

I **spettacoli** si terranno tutti a Piazza Duomo e saranno opportunamente distribuiti nelle fasce orarie disponibili.

Essi inoltre potranno avere un numero limitato di **spettatori** per evitare assembramenti, dunque sarà importante, per chi intende vederli, **prenotare** con cura e con anticipo lo **spettacolo** a cui si vorrebbe più voler **partecipare**.

La prenotazione per un **evento** è singola per permettere a tutti di poter partecipare ad almeno uno **spettacolo** tra quelli proposti.

Tuttavia, per evitare che alcuni **spettacoli** possano rimanere con **posti** liberi (a causa della limitazione ad una sola **prenotazione** per **persona**), sarà possibile a chiunque, solo per un'ora prima dell'inizio dello spettacolo, potersi **prenotare**, violando temporaneamente il limite imposto.

Progettazione Concettuale

Specifiche sui dati


Si vuole progettare un sistema informativo che permetta una miglior organizzazione del Carnevale di Acireale. Possiamo trovare diverse parti necessarie per realizzarlo.

In particolare, possiamo individuare le **persone/clienti** (circa 50000), per i quali rappresenteremo alcuni dati anagrafici quali il **numero di telefono, nome, cognome, codice fiscale** e la **data di nascita**.

Ogni persona per poter **acquistare** uno o più **biglietti** dovrà prima essere registrata nel sistema, quindi dovrà avere un **account** dove poter aggiungere le informazioni personali o anche quelle di altri membri familiari.

Ogni **account**, oltre ad avere associate le persone ad esso, avrà un'**email** che dovrà essere univoca nel sistema, e una **password** (salvata nel sistema con una funzione hash), che saranno usate per effettuare il login nel sistema.

I **biglietti** potranno appartenere ad una sola persona ed avranno un **codice univoco** (generato tramite una funzione hash) che sarà convertito in un **codice QR** per essere velocemente scannerizzato nelle entrate al circuito. Inoltre i biglietti saranno validi solo per un giorno.



I **spettacoli** saranno caratterizzati da un **nome** (che sarà univoco per spettacolo), dalla **data e ora di inizio**, dalla **durata** e infine da una dettagliata **descrizione** di quest'ultimo.

Gli **spettatori** per poter partecipare ai **spettacoli** dovranno essersi prenotati. Ogni **persona** può fare solo una prenotazione al giorno, tuttavia è possibile prenotarsi a più spettacoli se quest'ultimi hanno ancora posti disponibili un'ora prima del loro inizio. I **posti disponibili** saranno per tutti gli spettacoli gli stessi e saranno calcolati per garantire al meglio la distanza minima di sicurezza tra le persone. (Per questo progetto useremo un numero esemplificativo di massimo **1000 posti**).

I **carri** si esibiranno all'interno delle **aree**, a differenza delle edizioni passate i carri staranno fermi in appositi punti all'interno delle aree e non circoleranno più (questo per garantire che le persone non stiano sempre ferme all'interno della stessa area e per migliorare i problemi di sicurezza con lo spostamento dei carri).

Gli **spettatori** avranno la possibilità di **commentare** e **votare** i carri tramite il proprio **account** e grazie a questo sarà possibile stabilire una **graduatoria** dei migliori carri del Carnevale.

Un'**area** è caratterizzata da un **ID** che la identifica, **indirizzo**, **numero civico** dal quale inizia l'area e da un **limite di persone** che essa può accettare in circolazione (stabilito in base alla grandezza dell'area stessa). Ci sarà anche un'area "speciale" chiamata fuori circuito in cui circoleranno le persone non ancora entrate nel circuito.

Specifiche sulle Operazioni

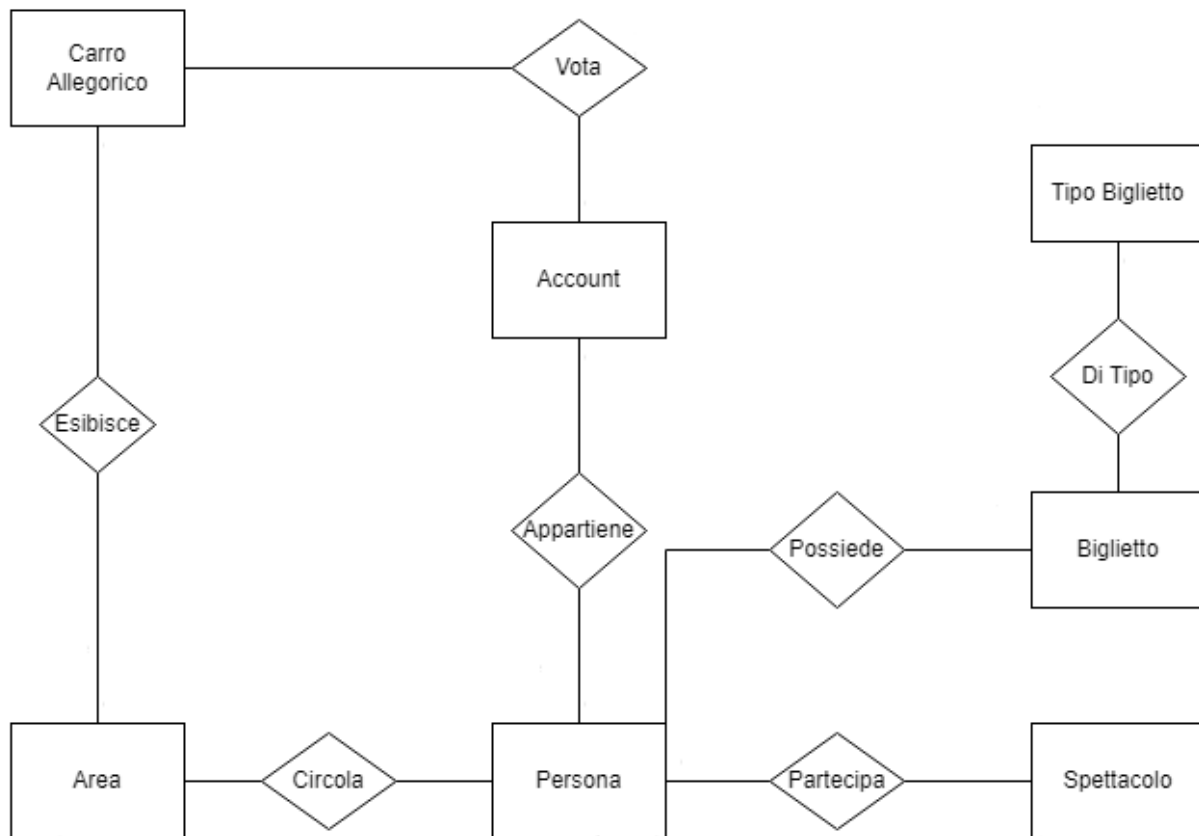
Le principali operazioni sui dati potrebbero essere:

- 1. Aggiunta di un nuovo account**
- 2. Aggiunta di una nuova persona**
- 3. Cambio di Area di una persona**
- 4. Prenotazione ad uno spettacolo**
- 5. Cancellazione prenotazione**
- 6. Votazione carro**
- 7. Importare denaro nell'account**
- 8. Acquisto biglietto**
- 9. Visualizzare il numero di persone in circola in un'area**
- 10. Visualizzare informazioni sui Carri**
- 11. Visualizzare l'incasso totale**
- 12. Visualizzare il numero di prenotati agli spettacoli**

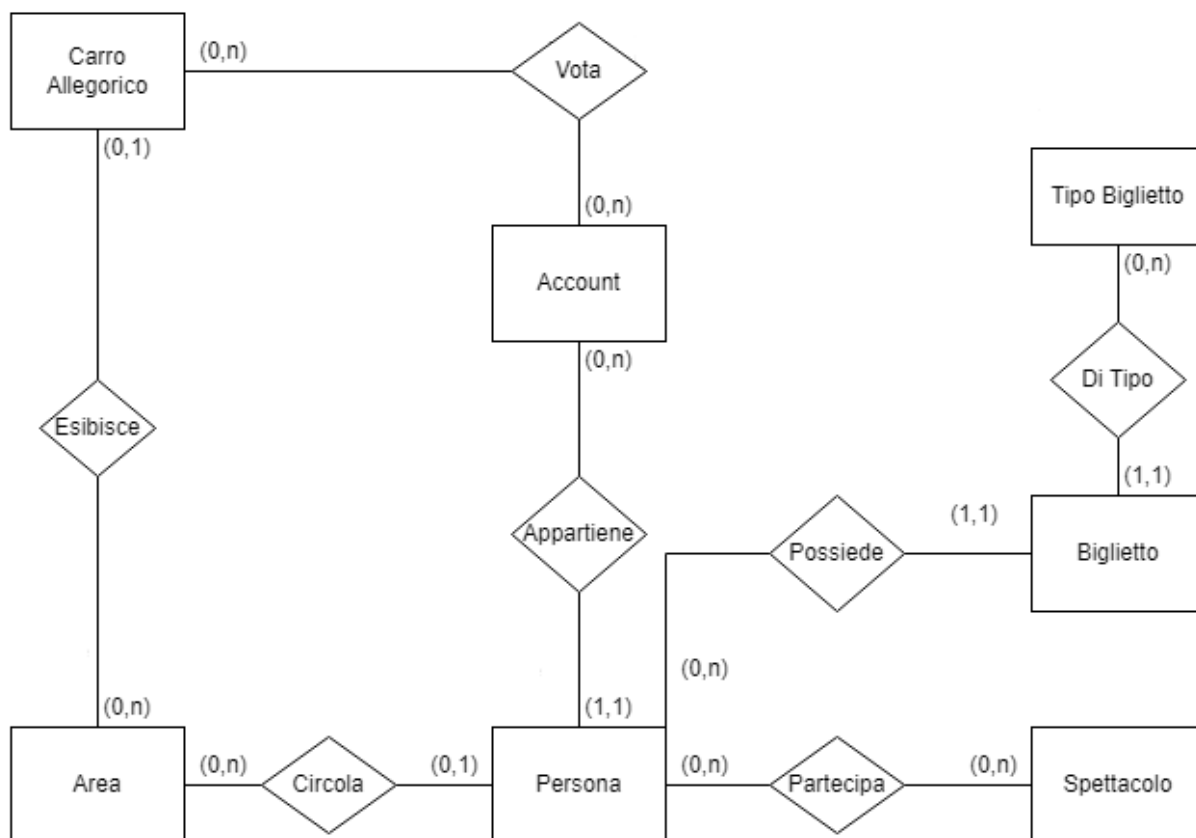
Analisi dei Requisiti

| Termine | Descrizione | Sinonimi | Termini collegati |
|------------------|---|-----------------------|-----------------------------|
| Account | Account contenente una o più persone | Profilo | Biglietto, Carro Allegorico |
| Persona | Individuo che è iscritto al sistema e intende partecipare al Carnevale | Cliente, Partecipante | Area, Spettacolo, Account |
| Biglietto | Oggetto necessario per poter entrare dentro il circuito del Carnevale | Ticket | Persona |
| Area | Piccole/medie zone presenti all'interno del circuito create per poter evitare al meglio gli assembramenti | | Persona, Carro Allegorico |
| Spettacolo | Evento di intrattenimento tenuto in un palco situato a Piazza Duomo | Evento, Concerto | Persona |
| Carro Allegorico | Un carro realizzato in cartapesta che raggiunge altezze e dimensioni grandissime e che, grazie ad alcuni meccanismi, è in grado anche di muoversi durante la sua esibizione | Carro di Carnevale | Area, Persona |

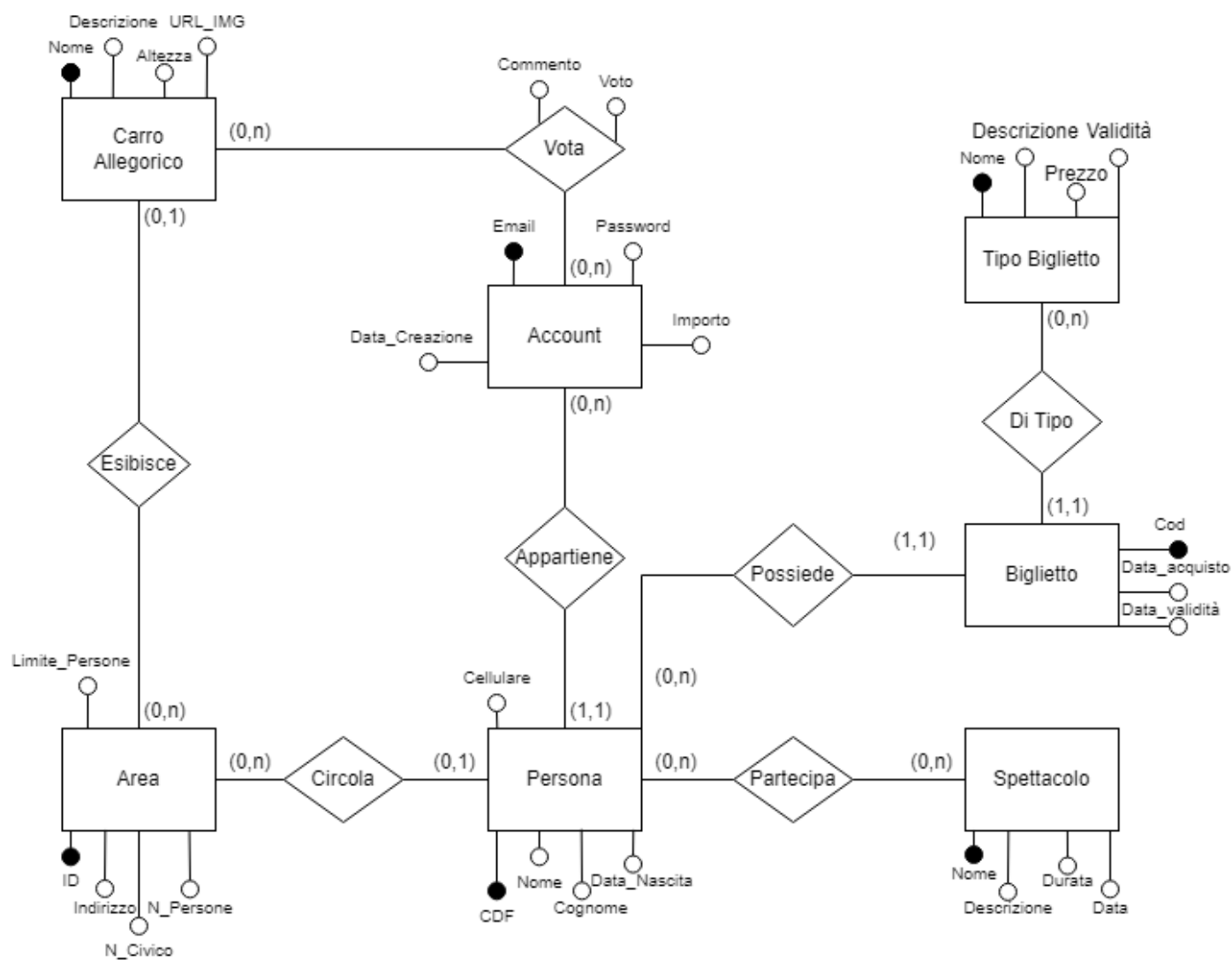
Schema scheletro



Schema intermedio (aggiunta cardinalità associazioni)



Schema finale (Aggiunti tutti gli attributi)



Vincoli non esprimibili dal modello E/R

- Un account può votare un carro solamente una volta
- Ogni spettacolo può avere al massimo 1000 partecipanti
- Ogni persona può partecipare solo ad 1 spettacolo
- Una persona può partecipare a più spettacoli se e solo se nello spettacolo in cui intende partecipare ci sono ancora posti disponibili ad 1 ora prima del suo inizio
- Una persona non può circolare in una nuova area se in questa è stato raggiunto il limite massimo di persone

Dizionario dei dati (Entità)

| Entità | Descrizione | Attributi | Identificatore |
|------------------|---|--|----------------|
| Account | Account contenente una o più persone | ID, Email, Password, Data_Creazione, Importo | ID |
| Persona | Individuo che è iscritto al sistema e intende partecipare al Carnevale | CDF, Nome, Cognome, Cellulare, Data_Nascita | CDF |
| Biglietto | Oggetto utilizzato per poter prenotare gli spettacoli e poter entrare nelle aree | Cod, Data_acquisto, Data_validità | Cod |
| Area | Piccole/medie zone presenti all'interno del circuito create per poter evitare al meglio gli assembramenti | ID, Indirizzo, N_Civico, N_Persone, Limite_persone | ID |
| Spettacolo | Evento di intrattenimento tenuto in un palco situato a Piazza Duomo | Nome, Descrizione, Durata, Data | Nome |
| Carro Allegorico | Un carro realizzato in cartapesta che raggiunge altezze e dimensioni grandissime e che, grazie ad alcuni meccanismi, è in grado anche di muoversi | Nome, Descrizione, Altezza, URL_IMG | Nome |

| | | | |
|----------------|--|-------------------------------------|------|
| | durante la sua esibizione | | |
| Tipo Biglietto | Indica la tipologia del biglietto acquistato | Nome, Descrizione, Prezzo, Validità | Nome |

Dizionario dei dati (Relazioni)

| Relazione | Entità partecipanti | Descrizione | Attributi |
|------------|---------------------------|--|----------------|
| Appartiene | Persona, Account | Associa ogni persona con il proprio account | |
| Possiede | Account, Biglietto | Associa i biglietti con gli account delle persone | |
| Partecipa | Persona, Spettacolo | Associa le persone partecipanti ad uno (o più) spettacoli | |
| Circola | Persona, Area | Serve a tracciare in quale area le persone stanno circolando | |
| Esibisce | Carro Allegorico, Area | Permette di sapere in quale area si trovano i Carri allegorici | |
| Vota | Account, Carro Allegorico | Tiene traccia dei commenti e votazioni degli account ai carri | Commento, Voto |
| Di Tipo | Biglietto, Tipo Biglietto | Associa il tipo di biglietto al biglietto | |

Progettazione logica

Tavola dei volumi

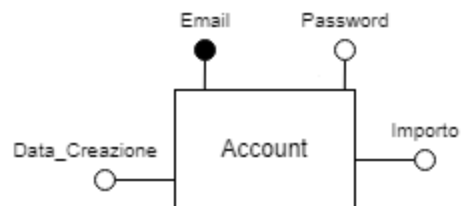
| Concetto | Tipo | Volume |
|------------------|------|---------|
| Account | E | 30'000 |
| Persona | E | 50'000 |
| Biglietto | E | 100'000 |
| Tipo Biglietto | E | 2 |
| Spettacolo | E | 30 |
| Area | E | 6 |
| Carro Allegorico | E | 12 |
| Esibisce | R | 12 |
| Circola | R | 30'000 |
| Vota | R | 200'000 |
| Appartiene | R | 50'000 |
| Possiede | R | 100'000 |
| Partecipa | R | 30'000 |
| Di Tipo | R | 100'000 |

Tavola delle Frequenze

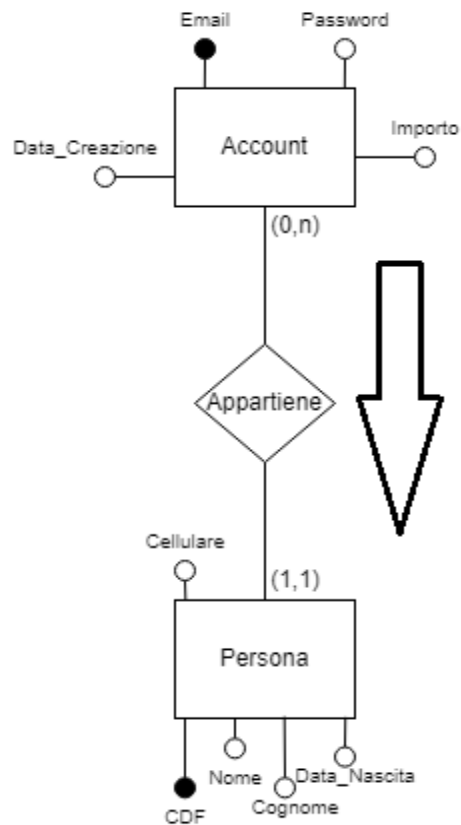
| Operazione | Descrizione | Frequenza |
|------------|---|----------------|
| Op.1 | Aggiunta di un nuovo account | 3'000/giorno |
| Op.2 | Aggiunta di una nuova persona | 7'000/giorno |
| Op.3 | Cambio di Area di una persona | 320'000/giorno |
| Op.4 | Prenotazione ad uno spettacolo | 5'000/giorno |
| Op.5 | Cancellazione prenotazione | 500/giorno |
| Op.6 | Votazione carro | 40'000/giorno |
| Op.7 | Importare denaro nell'account | 4'000/giorno |
| Op.8 | Acquisto biglietto | 5'000/giorno |
| Op.9 | Visualizzare il numero di persone in circolazione in un'area | 320'000/giorno |
| Op.10 | Visualizzare i dettagli di un carro | 200'000/giorno |
| Op.11 | Visualizzare il guadagno totale | 3/mese |
| Op.12 | Visualizzare il numero di prenotati agli spettacoli | 7/giorno |

Schemi delle Operazioni

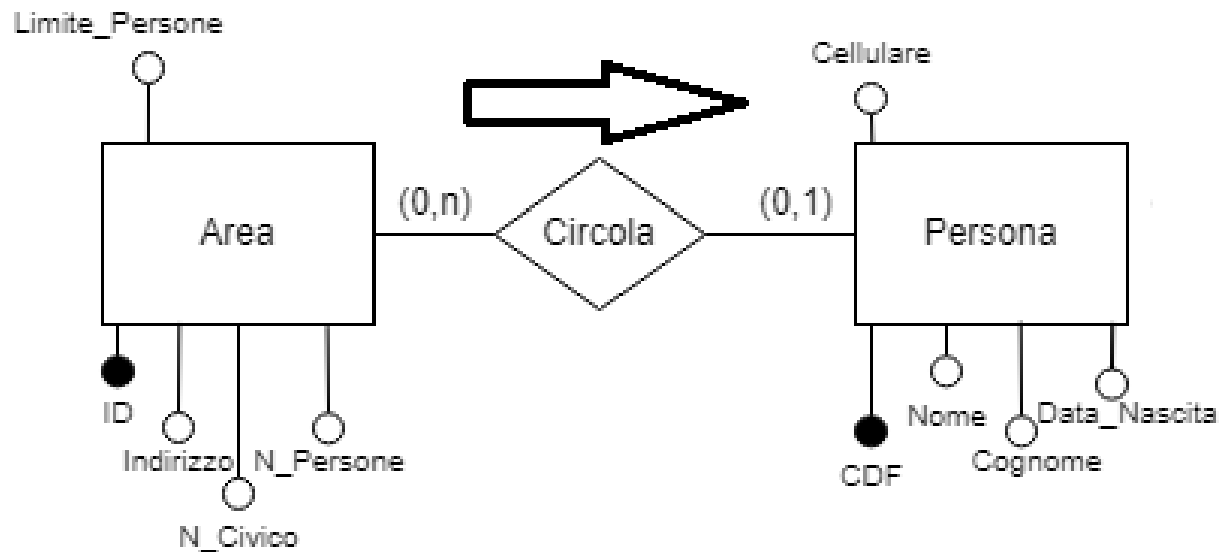
Op.1 - Aggiunta di un nuovo account



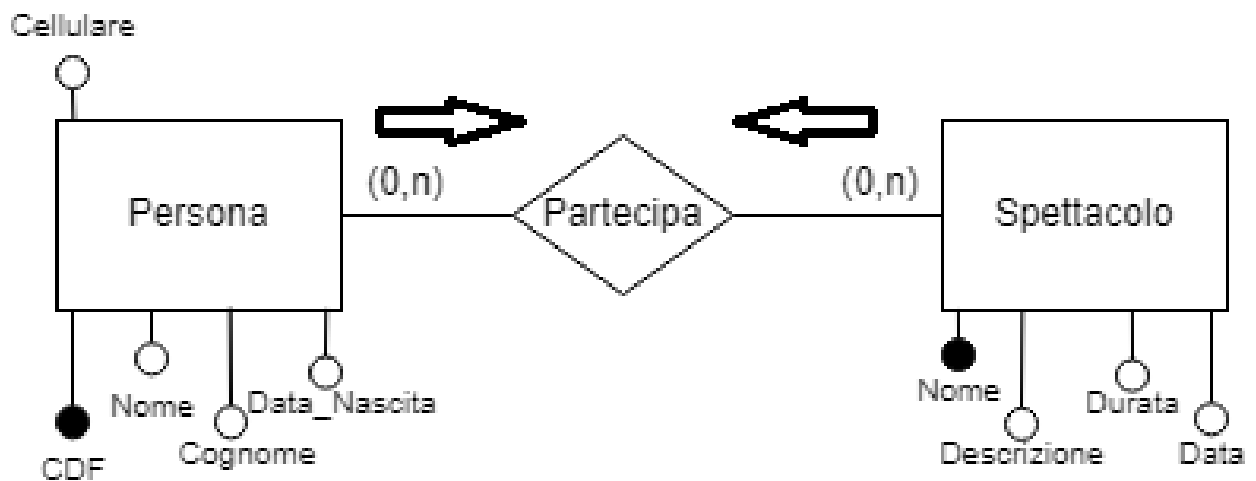
Op.2 - Aggiunta di una nuova persona



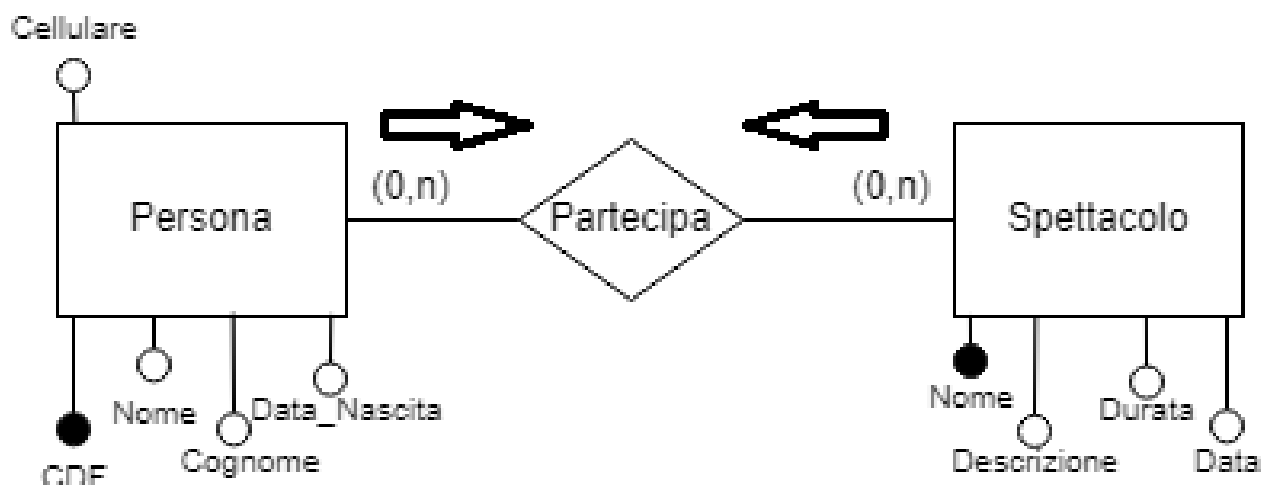
Op.3 - Cambio di Area di una persona



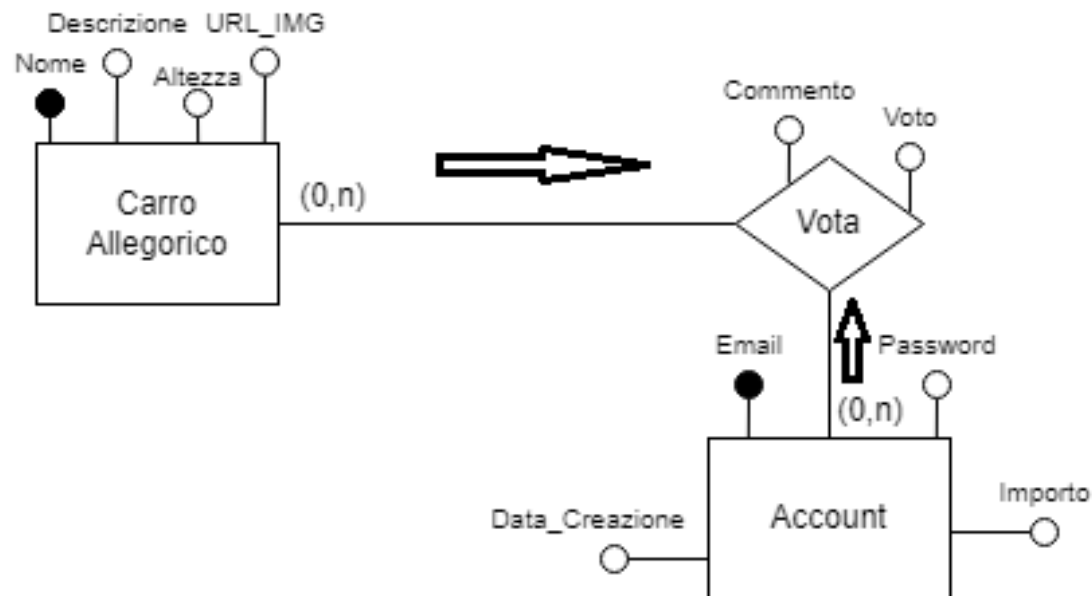
Op.4 - Prenotazione ad uno spettacolo



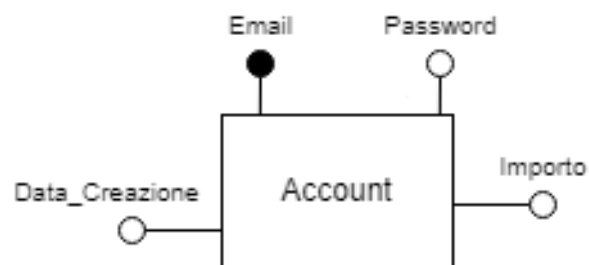
Op.5 - Cancellazione prenotazione



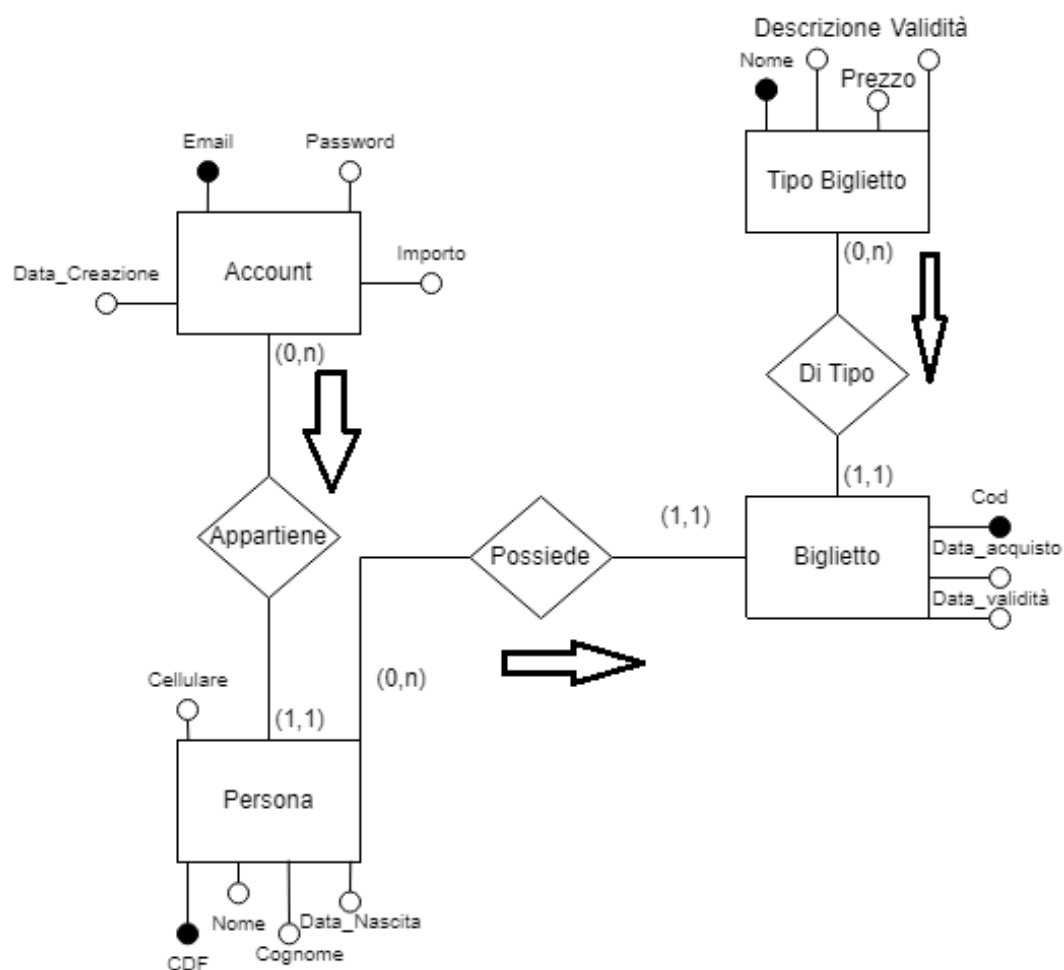
Op.6 - Votazione carro



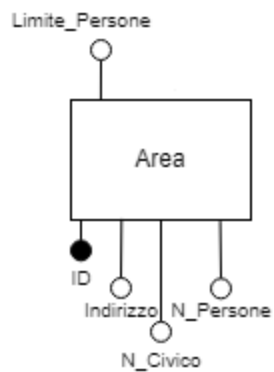
Op.7 - Importare denaro nell'account



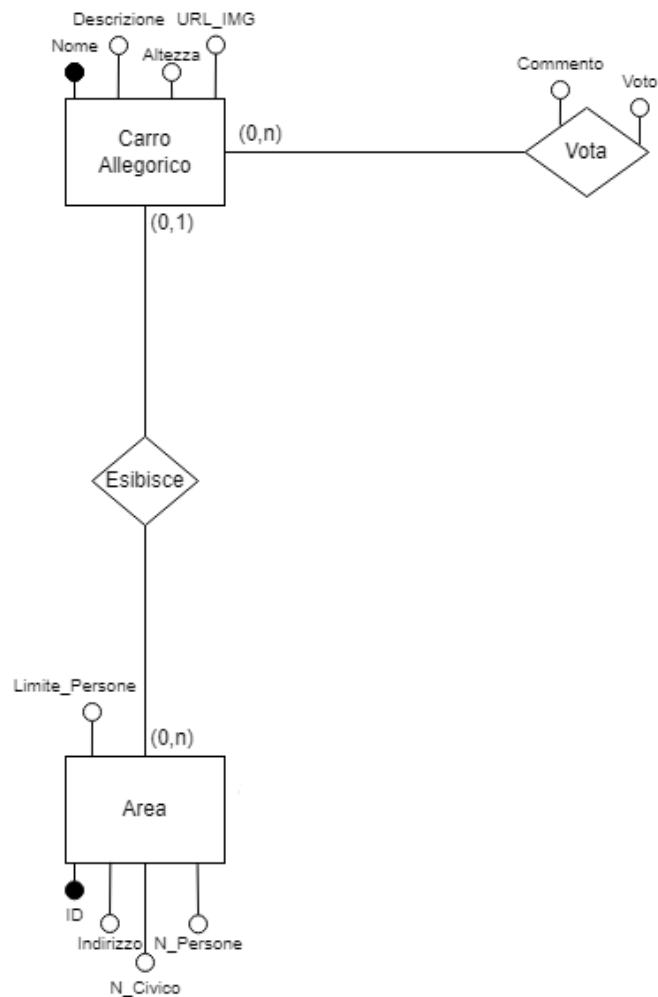
Op.8 - Acquisto biglietto



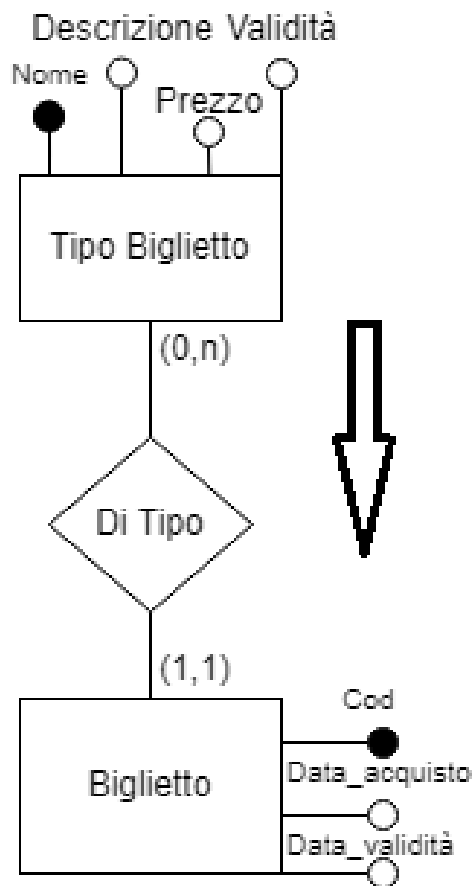
Op.9 - Visualizzare il numero di persone in circolazione in un'area



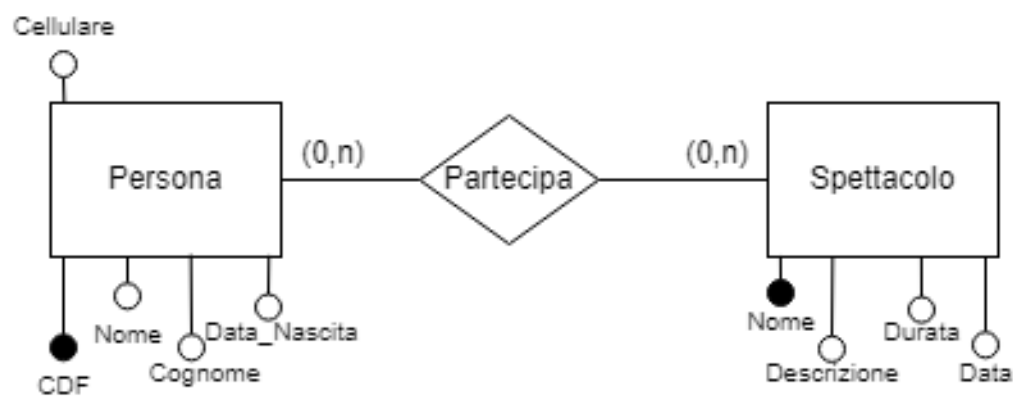
Op.10 - Visualizzare informazioni sui carri



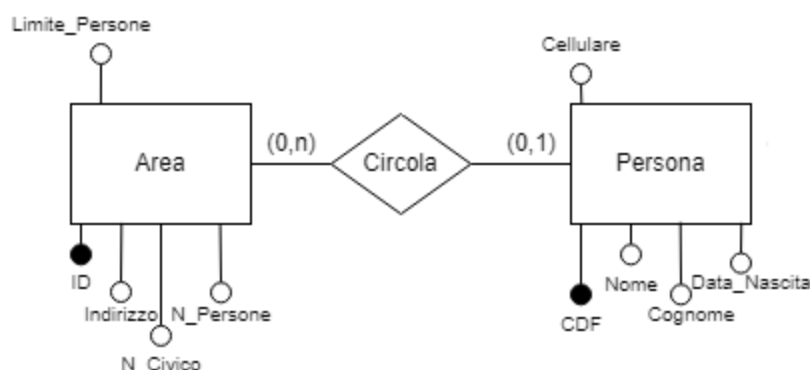
Op.11 - Visualizzare l'incasso totale



Op.12 - Visualizzare il numero di prenotati agli spettacoli



Valutazione costo ridondanza attributo “N_Persone” in Area



Le operazioni coinvolte sono il (Op.3) **Cambio di area di una persona** e (Op.9) **Visualizzare il numero di persone in circolazione in un'area**.

Valutazione costi con la ridondanza

Per l'Op.3 si ha:

- 1 accesso in lettura per cercare l'area attuale (vecchia)
- 1 accesso in scrittura per diminuire di 1 N_Persone
- 1 accesso in scrittura a Circola per cambiare l'area
- 1 accesso in lettura per trovare la nuova area
- 1 accesso in scrittura per incrementare di 1 N_Persone

Lecture totali Op.3 = $(2 + 3 \cdot 2) \cdot 320'000 = 2'560'000$ letture al giorno

Per l'Op.9 si ha:

- 1 accesso in lettura ad Area

Lecture totali Op.9 = $1 \cdot 320'000 = 320'000$ letture al giorno

In totale si hanno 2'880'000 letture al giorno con ridondanza

Valutazione costi senza ridondanza

Per l'Op.3 si ha:

- 1 accesso in scrittura a Circola per cambiare l'area

Lettture totali Op.3 = $1 * 320'000 = 320'000$ letture al giorno

Per l'Op.9 si ha:

- 1 accesso in lettura ad Area
- 5'000 accessi (in media) in lettura a Circola

Lettture totali Op.9 = $5'001 * 320'000$ letture al giorno

In totale si hanno $5'001 * 320'000$ letture al giorno senza ridondanza

Da queste valutazioni concludiamo che è necessario mantenere la ridondanza.

Traduzioni delle associazioni molti a molti

Lo schema E/R comprende alcune associazioni molti a molti che verranno tradotte in entità al fine di ottenere associazioni uno a molti:

- L'associazione **Vota** viene trasformata nell'entità **Votazione** e sarà identificata dalla coppia degli identificativi di **Carro Allegorico** e **Account**
- L'associazione **Partecipa** viene trasformata nell'entità **Partecipante** e sarà identificata dalla coppia degli identificativi di **Persona** e **Spettacolo**

Traduzioni delle associazioni uno a molti

Lo schema E/R comprende alcune associazioni uno a molti che verranno inglobate dalle entità:

- L'associazione **Esibisce** tra **Carro Allegorico** e **Area** verrà inglobata dall'entità **Carro Allegorico**, l'entità **Carro Allegorico** avrà dunque un riferimento ad **Area**
- L'associazione **Circola** tra **Persona** e **Area** verrà inglobata dall'entità **Persona**, l'entità **Persona** avrà dunque un riferimento ad **Area**
- L'associazione **Appartiene** tra **Persona** e **Account** verrà inglobata dall'entità **Persona**, l'entità **Persona** avrà dunque un riferimento ad **Account**
- L'associazione **Possiede** tra **Persona** e **Biglietto** verrà inglobata dall'entità **Biglietto**, l'entità **Biglietto** avrà dunque un riferimento a **Persona**
- L'associazione **Di Tipo** tra **Biglietto** e **Tipo** **Biglietto** verrà inglobata dall'entità **Biglietto**, l'entità **Biglietto** avrà dunque un riferimento ad **Tipo Biglietto**

Schema logico

Persona(**CDF**, Cellulare, Nome, Cognome, Data_Nascita, FK_Area, FK_Account)

Area(**ID**, Indirizzo, N_Civico, N_Persone, Limite_Persone)

Carro Allegorico(**Nome**, Descrizione, Altezza, URL_IMG, FK_Area)

Account(**ID**, Email, Password, Data_Creazione, Importo)

Votazione(**FK Account, FK Carro**, Commento, Voto)

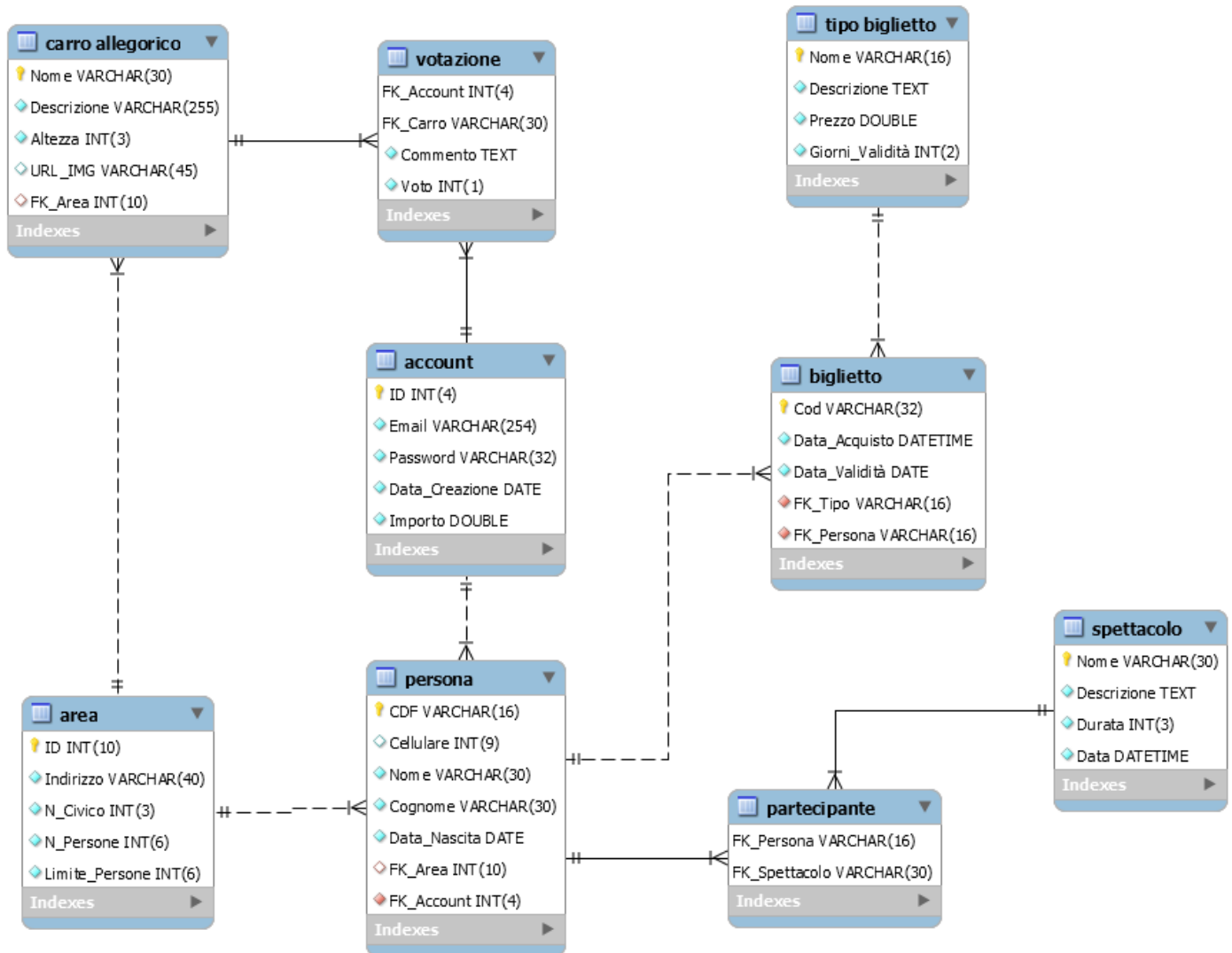
Partecipante(**FK Persona, FK Spettacolo**)

Spettacolo(**Nome**, Descrizione, Durata, Data)

Biglietto(**Cod**, Data_Acquisto, Data_Validità, FK_Tipo, FK_Persona)

Tipo Biglietto(**Nome**, Descrizione, Prezzo, Giorni_Validità)

Schema UML



Progettazione Fisica

Implementazione fisica della Base di Dati

```
CREATE TABLE `account` (  
  `ID` int(4) UNSIGNED NOT NULL,  
  `Email` varchar(254) NOT NULL,  
  `Password` varchar(32) NOT NULL,  
  `Data_Creazione` date NOT NULL DEFAULT current_timestamp(),  
  `Importo` double NOT NULL DEFAULT 0  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `area` (  
  `ID` int(10) UNSIGNED NOT NULL,  
  `Indirizzo` varchar(40) NOT NULL,  
  `N_Civico` int(3) NOT NULL,  
  `N_Persone` int(6) NOT NULL,  
  `Limite_Persone` int(6) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `biglietto` (  
  `Cod` varchar(32) NOT NULL,  
  `Data_Acquisto` datetime NOT NULL DEFAULT current_timestamp(),  
  `Data_Validità` date NOT NULL,  
  `FK_Tipo` varchar(16) NOT NULL,
```


```
`FK_Persona` varchar(16) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `carro allegorico` (  
  `Nome` varchar(30) NOT NULL,  
  `Descrizione` varchar(255) NOT NULL,  
  `Altezza` int(3) NOT NULL,  
  `URL_IMG` varchar(45) DEFAULT NULL,  
  `FK_Area` int(10) UNSIGNED DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `debug` (  
  `id` int(11) NOT NULL,  
  `message` varchar(255) DEFAULT NULL,  
  `time` timestamp NULL DEFAULT current_timestamp() ON UPDATE  
  current_timestamp()  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `partecipante` (  
  `FK_Persona` varchar(16) NOT NULL,  
  `FK_Spettacolo` varchar(30) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `persona` (  
  `CDF` varchar(16) NOT NULL,  
  `Cellulare` int(9) UNSIGNED DEFAULT NULL,
```

```
`Nome` varchar(30) NOT NULL,  
`Cognome` varchar(30) NOT NULL,  
`Data_Nascita` date NOT NULL,  
`FK_Area` int(10) UNSIGNED DEFAULT 0,  
`FK_Account` int(4) UNSIGNED NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `spettacolo` (  
  `Nome` varchar(30) NOT NULL,  
  `Descrizione` text NOT NULL,  
  `Durata` int(3) NOT NULL,  
  `Data` datetime NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `tipo biglietto` (  
  `Nome` varchar(16) NOT NULL,  
  `Descrizione` text NOT NULL,  
  `Prezzo` double NOT NULL,  
  `Giorni_Validità` int(2) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE `votazione` (  
  `FK_Account` int(4) UNSIGNED NOT NULL,  
  `FK_Carro` varchar(30) NOT NULL,  
  `Commento` text NOT NULL,  
  `Voto` int(1) NOT NULL
```



```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
ALTER TABLE `account`  
  ADD PRIMARY KEY (`ID`),  
  ADD UNIQUE KEY `Email` (`Email`);
```


```
ALTER TABLE `area`  
  ADD PRIMARY KEY (`ID`);
```

```
ALTER TABLE `biglietto`  
  ADD PRIMARY KEY (`Cod`),  
  ADD KEY `FK_Tipo` (`FK_Tipo`),  
  ADD KEY `FK_Persona` (`FK_Persona`);
```

```
ALTER TABLE `carro allegorico`  
  ADD PRIMARY KEY (`Nome`),  
  ADD KEY `FK_Area` (`FK_Area`);
```

```
ALTER TABLE `debug`  
  ADD PRIMARY KEY (`id`);
```

```
ALTER TABLE `partecipante`  
  ADD PRIMARY KEY (`FK_Persona`,`FK_Spettacolo`),  
  ADD KEY `FK_Persona` (`FK_Persona`),  
  ADD KEY `FK_Spettacolo` (`FK_Spettacolo`);
```



```
ALTER TABLE `persona`  
  ADD PRIMARY KEY (`CDF`),  
  ADD KEY `FK_Area` (`FK_Area`),  
  ADD KEY `FK_Account` (`FK_Account`);
```

```
ALTER TABLE `spettacolo`  
  ADD PRIMARY KEY (`Nome`);
```

```
ALTER TABLE `tipo biglietto`  
  ADD PRIMARY KEY (`Nome`);
```

```
ALTER TABLE `votazione`  
  ADD PRIMARY KEY (`FK_Account`, `FK_Carro`),  
  ADD KEY `FK_Account` (`FK_Account`),  
  ADD KEY `FK_Carro` (`FK_Carro`);
```

```
ALTER TABLE `account`  
  MODIFY `ID` int(4) UNSIGNED NOT NULL AUTO_INCREMENT;
```

```
ALTER TABLE `area`  
  MODIFY `ID` int(10) UNSIGNED NOT NULL AUTO_INCREMENT;
```

```
ALTER TABLE `debug`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```



```
ALTER TABLE `biglietto`
```

```
    ADD CONSTRAINT `biglietto_ibfk_1` FOREIGN KEY (`FK_Tipo`) REFERENCES  
`tipo biglietto` (`Nome`),
```

```
    ADD CONSTRAINT `biglietto_ibfk_2` FOREIGN KEY (`FK_Persona`)  
REFERENCES `persona` (`CDF`);
```

```
ALTER TABLE `carro allegorico`
```

```
    ADD CONSTRAINT `carro allegorico_ibfk_1` FOREIGN KEY (`FK_Area`)  
REFERENCES `area` (`ID`);
```

```
ALTER TABLE `partecipante`
```

```
    ADD CONSTRAINT `partecipante_ibfk_1` FOREIGN KEY (`FK_Persona`)  
REFERENCES `persona` (`CDF`),
```

```
    ADD CONSTRAINT `partecipante_ibfk_2` FOREIGN KEY (`FK_Spettacolo`)  
REFERENCES `spettacolo` (`Nome`);
```

```
ALTER TABLE `persona`
```

```
    ADD CONSTRAINT `persona_ibfk_1` FOREIGN KEY (`FK_Area`) REFERENCES  
`area` (`ID`),
```

```
    ADD CONSTRAINT `persona_ibfk_2` FOREIGN KEY (`FK_Account`)  
REFERENCES `account` (`ID`);
```

```
ALTER TABLE `votazione`
```

```
    ADD CONSTRAINT `votazione_ibfk_2` FOREIGN KEY (`FK_Carro`) REFERENCES  
`carro allegorico` (`Nome`),
```

```
    ADD CONSTRAINT `votazione_ibfk_3` FOREIGN KEY (`FK_Account`)  
REFERENCES `account` (`ID`);
```

Implementazione delle operazioni

Op.1 - Aggiunta di un nuovo Account

```
INSERT INTO `account` (`ID`, `Email`, `Password`, `Data_Creazione`,  
`Importo`) VALUES (NULL, '<Email>', '<Password>',  
current_timestamp(), '<Importo>');
```

Op.2 - Aggiunta di una nuova persona

```
INSERT INTO `persona` (`CDF`, `Cellulare`, `Nome`, `Cognome`,  
`Data_Nascita`, `FK_Area`, `FK_Account`) VALUES ('<CDF>',  
'<Cellulare>', '<Nome>', '<Cognome>', '<Data>', '<ID_Area>',  
'ID_Account');
```

Op.3 - Cambio di Area di una Persona

```
UPDATE `persona` SET `FK_Area` = '<ID_Area>' WHERE  
`persona`.`CDF` = '<CDF>';
```

Op.4 - Prenotazione ad uno Spettacolo

```
INSERT INTO `partecipante` (`FK_Persona`, `FK_Spettacolo`)  
VALUES ('<CDF>', '<Nome spettacolo>');
```

Op.5 - Cancellazione prenotazione

```
DELETE FROM `partecipante` WHERE `FK_Persona` = '<CDF>';
```

Op.6 - Votazione Carro

```
INSERT INTO `votazione` (`FK_Account`, `FK_Carro`,  
`Commento`, `Voto`) VALUES ('<ID_Account>', 'Nome carro',  
'<Commento>', '<Voto>');
```

Op.7 - Importare denaro nell'Account

```
UPDATE `account` SET `Importo` = `Importo` + '<Importo>'  
WHERE `account`.`ID` = <ID_Account>;
```

Op.8 - Acquisto biglietto

```
INSERT INTO `biglietto` (`Cod`, `Data_Acquisto`,  
`Data_Validità`, `FK_Tipo`, `FK_Persona`) VALUES ('<Cod>',  
current_timestamp(), '<Data_Validità>', '<FK_TIPO>',  
'<CDF_Persona>');
```

Op.9 - Visualizzare il numero di persone in circolazione in un'area

```
SELECT `N_Persone` FROM `area` WHERE `ID` = <ID_Area>;
```

Op.10 - Visualizzare informazioni sui Carri

```
SELECT `carro allegorico`.`Nome`, `carro  
allegorico`.`Descrizione`, `carro allegorico`.`Altezza`,  
AVG(`votazione`.`Voto`) AS 'Voto medio', `area`.`Indirizzo`,  
`area`.`N_Civico` FROM `carro allegorico`, `votazione`, `area`  
WHERE `votazione`.`FK_Carro` = `carro allegorico`.`Nome` AND  
`area`.`ID` = `carro allegorico`.`FK_Area` GROUP BY `carro  
allegorico`.`Nome` ORDER BY `Voto medio` DESC;
```

Op.11 - Visualizzare l'incasso totale

```
SELECT sum(`tipo biglietto`.`Prezzo`) as 'Incasso Totale' FROM  
`biglietto`,`tipo biglietto` WHERE `biglietto`.`FK_Tipo`=`tipo  
biglietto`.`Nome`;
```

Op.12 - Visualizzare il numero di partecipanti agli spettacoli

```
SELECT `spettacolo`.`Nome`,  
COUNT(`partecipante`.`FK_Persona`) AS 'N_Partecipanti' FROM  
`partecipante` RIGHT JOIN `spettacolo` ON  
`spettacolo`.`Nome` = `partecipante`.`FK_Spettacolo` GROUP  
BY `spettacolo`.`Nome` ORDER BY `N_Partecipanti` DESC;
```

Implementazione dei Trigger

Trigger n°1 - Entrata in un'area

```
CREATE TRIGGER `Entrata in un'area` BEFORE UPDATE ON `persona`  
FOR EACH ROW BEGIN
```

```
    IF ((SELECT `area`.`N_Persone` FROM `area` WHERE `area`.`ID` =  
NEW.`FK_Area`) + 1 <= (SELECT `area`.`Limite_Persone` FROM `area`  
WHERE `area`.`ID` = NEW.`FK_Area`)) THEN UPDATE `area` SET  
`area`.`N_Persone` = `area`.`N_Persone`+1 WHERE `area`.`ID` =  
NEW.`FK_Area`;
```

```
    UPDATE `area` SET `area`.`N_Persone` = `area`.`N_Persone`-1  
WHERE `area`.`ID` = OLD.`FK_Area`;
```

```
    ELSE
```

```
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Limite persone  
raggiunto';
```

```
    END IF;
```

```
END
```


Trigger n°2 - Nuova persona

```
CREATE TRIGGER `Nuova Persona` BEFORE INSERT ON `persona`  
FOR EACH ROW BEGIN  
    IF ((SELECT `area`.`N_Persone` FROM `area` WHERE `area`.`ID` =  
'0') + 1 <= (SELECT `area`.`Limite_Persone` FROM `area` WHERE  
`area`.`ID` = '0')) THEN UPDATE `area` SET `area`.`N_Persone` =  
`area`.`N_Persone`+1 WHERE `area`.`ID` = '0';  
  
    ELSE  
  
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Limite persone  
raggiunto';  
  
    END IF;  
END
```

Trigger n°3 - Acquisto del Biglietto

```
CREATE TRIGGER `Acquisto del biglietto` BEFORE INSERT ON
`biglietto` FOR EACH ROW BEGIN
DECLARE prezzo_biglietto integer;
DECLARE ID_ACCOUNT integer;
SELECT `tipo biglietto`.`Prezzo` INTO `prezzo_biglietto` FROM `tipo
biglietto` WHERE `tipo biglietto`.`Nome` = NEW.`FK_Tipo`;
SELECT `account`.`ID` INTO `ID_ACCOUNT` FROM `account` JOIN
`persona` ON `account`.`ID`=`persona`.`FK_Account` WHERE
`persona`.`CDF` = NEW.`FK_Persona`;
    IF((SELECT `account`.`Importo` FROM `account` WHERE
`account`.`ID`=ID_ACCOUNT) > prezzo_biglietto) THEN
        UPDATE `account` SET `account`.`Importo` = `account`.`Importo`
- prezzo_biglietto WHERE `account`.`ID`=ID_ACCOUNT;
    ELSE
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Soldi non
sufficienti';
    END IF;
END
```

Trigger n°4 - Controllo prenotazione a spettacolo

```
CREATE TRIGGER `Controllo prenotazione` BEFORE INSERT ON
`partecipante` FOR EACH ROW BEGIN
  Declare giorno_spettacolo datetime;
  Declare giorno_attuale datetime;
  DECLARE posti_occupati integer;
  SELECT COUNT(`partecipante`.`FK_Persona`) INTO `posti_occupati`
  FROM `partecipante` RIGHT JOIN `spettacolo` ON
  `spettacolo`.`Nome` = NEW.`FK_Spettacolo`;
  SELECT `spettacolo`.`Data` INTO `giorno_spettacolo` FROM
  `spettacolo` WHERE `spettacolo`.`Nome` = NEW.`FK_Spettacolo`;
  SET giorno_attuale = NOW();
  IF(posti_occupati > 1000) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Posti finiti';
  ELSE
    IF(SELECT COUNT(*) FROM `partecipante` WHERE
    `partecipante`.`FK_Persona` = NEW.`FK_Persona` > 0 AND
    (TIMESTAMPDIFF(HOUR, giorno_attuale, giorno_spettacolo) > 1 OR
    TIMESTAMPDIFF(MONTH, giorno_attuale, giorno_spettacolo) > 1 OR
    TIMESTAMPDIFF(YEAR, giorno_attuale, giorno_spettacolo) > 1))
    THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'La persona è già
      prenotata ad uno spettacolo';
    END IF;
  END IF;
END
```

Trigger n°5 - “Prenotazione - Controllo biglietto”

```
CREATE TRIGGER `Prenotazione - Controllo biglietto` BEFORE
INSERT ON `partecipante` FOR EACH ROW BEGIN
DECLARE giorni_validità integer;
DECLARE inizio_validità datetime;
DECLARE giorno_spettacolo datetime;
DECLARE `done` BOOL DEFAULT FALSE;
DECLARE `trovato` BOOL DEFAULT FALSE;
DECLARE `cur` CURSOR FOR
SELECT `biglietto`.`Data_Validità`, `tipo biglietto`.`Giorni_Validità`
FROM `biglietto` JOIN `tipo biglietto` ON `biglietto`.`FK_Tipo` = `tipo
biglietto`.`Nome` WHERE `biglietto`.`FK_Persona` =
NEW.`FK_Persona`;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET `done` := TRUE;
SET `trovato` = false;
SELECT `spettacolo`.`Data` INTO `giorno_spettacolo` FROM
`spettacolo` WHERE `spettacolo`.`Nome`=NEW.`FK_Spettacolo`;
OPEN `cur`;
    `read_loop`:LOOP
    FETCH `cur` INTO `inizio_validità`, `giorni_validità`;
    IF `done` or `trovato` THEN
        CLOSE `cur`;
        LEAVE `read_loop`;
    END IF;
```

```
IF ((TIMESTAMPDIFF(DAY,inizio_validità,giorno_spettacolo) <
giorni_validità) AND
TIMESTAMPDIFF(MONTH,inizio_validità,giorno_spettacolo) = 0 AND
inizio_validità < giorno_spettacolo) THEN
    SET `trovato` = true;
END IF;

/* insert into `debug`(message) VALUES (concat('inizio validità:
',inizio_validità,' giorni validità: ',giorni_validità,' giorno spettacolo:
',giorno_spettacolo,' differenza giorni con lo spettacolo:
',TIMESTAMPDIFF(DAY,inizio_validità,giorno_spettacolo), ' trovato:
',trovato,' risultato dell if ',
(TIMESTAMPDIFF(DAY,inizio_validità,giorno_spettacolo) <
giorni_validità))); */ /*DEBUGGING*/

END LOOP;

IF (!`trovato`) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Biglietto
valido non trovato per lo spettacolo';
END IF;

END
```