

Raffinamento dei Requisiti:

1. requisiti Opere:

- 1.1 nome : stringa
- 1.2 categoria, una tra:
  - 1.2.1 dipinto,
  - 1.2.2 scultura,
  - 1.2.3 mosaico,
  - 1.2.4 manoscritto,
- 1.3 autore (v. req. 2)
- 1.4 anno\_realizzazione
- 1.4 tecnica : stringa
- 1.5 correnti\_artistica : stringa

2. requisiti sugli Autori:

- 2.1 nome\_arte : stringa
- 2.2 luogo (v. req. 6)
- 2.3 data\_nascita
- 2.4 data\_morte
- 2.5 opere\_realizzate (v. req. 1)

3. requisiti Esposizione

- 3.1 stato\_esposizione, uno tra:
  - 3.1.1 permanenti
  - 3.1.2 temporanei (v. req. 4)

4. requisiti Esposizione Temporanea:

- 4.1 nome : stringa
- 4.2 prezzo di accesso
- 4.3 intervallo\_tempo
  - 4.3.1 data\_inizio
  - 4.3.2 data\_fine

5. requisiti Tariffe:

- 5.1 nome
- 5.2 prezzo base (reale  $\geq 0$ )

6. requisiti Visitatori

- 6.1 tariffa (v. req. 5)
- 6.2 istante\_venduto (data inizio)
- 6.2 data\_validita (data del giorno della mostra)

7. requisiti biglietti:

- 7.1 tipo, uno tra:
  - 7.1.1 standard
    - 7.1.1.1 esposizioni permanenti (v.req 3.1.1)
    - 7.1.1.2 tariffa base (v. req. 5)
  - 7.1.2 extended access
    - 7.1.2.1 esposizioni permanenti (v.req 3.1.1)
    - 7.1.2.1 esposizioni temporanee (v. req. 4)
    - 7.1.2.3 prezzo (prezzo base tariffa + prezzo accesso opere temporanee)

## 6. Requisiti sui luoghi

6.1. città

6.2. regione

6.3. nazione

999. il sistema deve offrire le seguenti funzionalità

999.0 gli utenti sono responsabili di gestione delle opere e delle esposizioni

999.1 inserimento e rimozione di opere d'arte;

999.2 creazione di un'esposizione, con le opere relative;

999.3 aggiunta di un artista e modifica delle sue informazioni;

999.4 modifica delle informazioni di un'esposizione;

999.5 registrazione della vendita di uno o più biglietti di una certa tipologia e validi per una certa data.

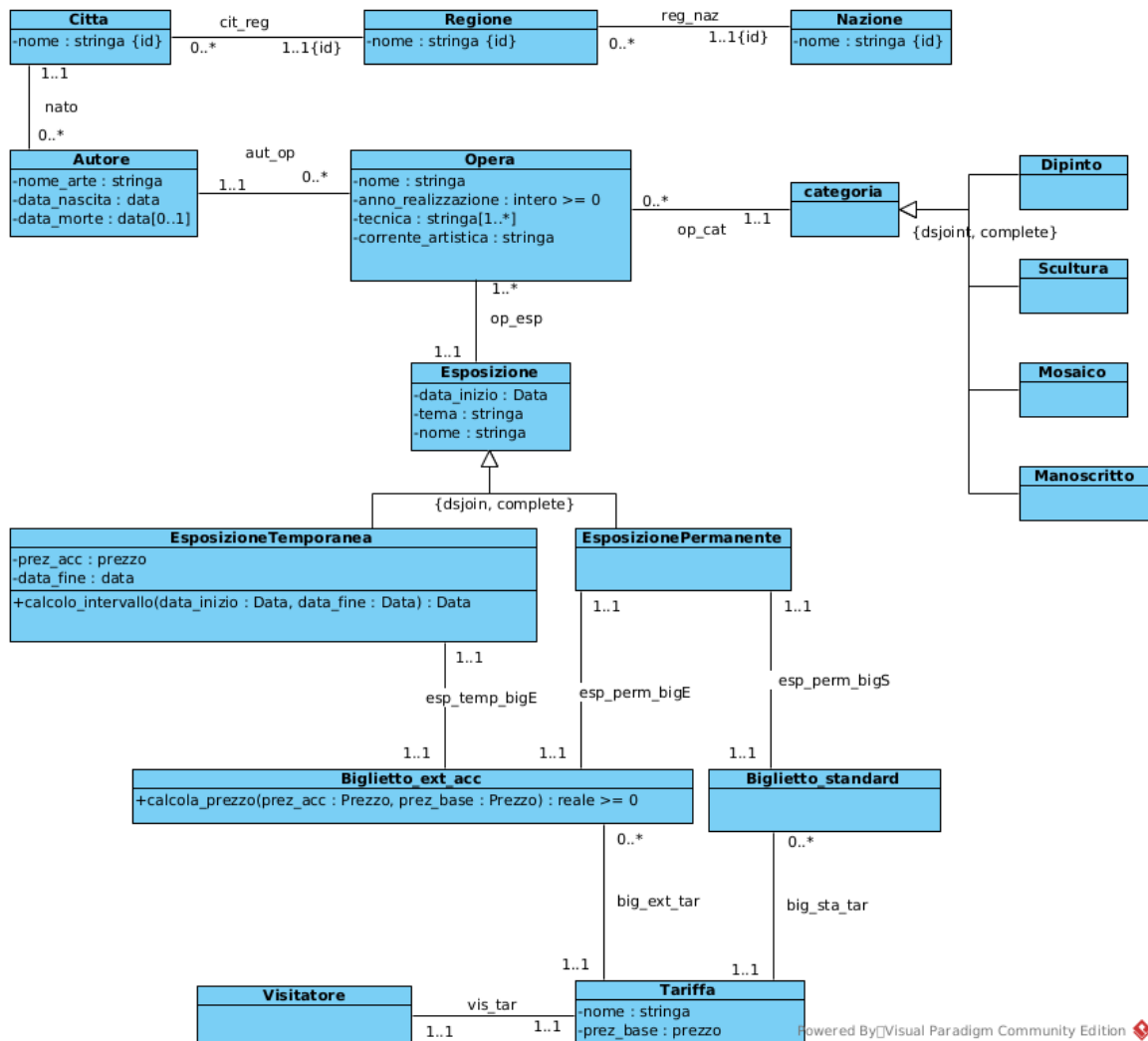
999.6 calcolo degli introiti totali per una certa esposizione temporanea, in un dato periodo. (Solo e unicamente per le funzionalità di seguito, dare la specifica completa.)

999.7 Data un'esposizione temporanea e una data, calcolare il numero di biglietti Extended access venduti per tale esposizione e validi per tale data.

999.8 Dato un periodo di tempo, calcolare le esposizioni temporanee per la quale è stato venduto il più alto numero di biglietti Extended access nel periodo.

999.9 Dato un artista e un periodo di tempo, calcolare le opere realizzate dall'artista che non sono mai state esposte nel periodo dato.

## Diagramma UML concettuale - ANALISI



---

Specifica dei tipi di Dato:

Specifica del tipo di dato:

Prezzo : Reale  $\geq 0$

---

Specifica di Classi, Vincoli e operazioni:

Classe Autore

[V.Autore.dateCoerenti]

La data di morte, se presente, deve essere successiva alla data di nascita.

Per ogni a:Autore, se a.data\_morte esiste, allora a.data\_nascita < a.data\_morte.

Non esiste a:Autore tale che a.data\_morte esiste e a.data\_morte  $\leq$  a.data\_nascita.

Classe Opera

[V.Opera.annoValido]

L'anno di realizzazione di un'opera deve essere maggiore o uguale a 0.

Per ogni o:Opera, o.anno\_realizzazione  $\geq 0$ .

[V.Opera.unicitaNomeAutore]

Due opere dello stesso autore non possono avere lo stesso nome.

[V.Opera.esposizione]

Un'opera non può essere esposta contemporaneamente in due esposizioni diverse

Classe Esposizione

[V.Esposizione.dateCoerenti]

Per ogni e:EsposizioneTemporanea, la data di fine deve essere successiva alla data di inizio.

Per ogni e:EsposizioneTemporanea, e.data\_inizio < e.data\_fine.

Non esiste e:EsposizioneTemporanea tale che e.data\_inizio  $\geq$  e.data\_fine.

[V.Esposizione.tipologia]

Ogni esposizione deve essere o temporanea o permanente.

Per ogni e:Esposizione, e è istanza di EsposizioneTemporanea oppure di EsposizionePermanente, ma non di entrambe.

[O.EsposizioneTemporanea.calcolo\_intervallo]

calcola\_intervallo(data\_inizio: Data, data\_fine: Data): data

Precondizione

prezz\_acc  $\geq 0$

prezz\_base  $\geq 0$

Postcondizione

Restituisce un valore reale maggiore o uguale a zero:

risultato = prezz\_base + prezz\_acc

Classe Tariffa

[V.Tariffa.prezzoBaseValido]

Il prezzo base deve essere maggiore o uguale a zero.

Per ogni t:Tariffa, t.prezzo\_base  $\geq 0$ .

Classe Biglietto

[V.Biglietto.dataValidita]

La data di validità di un biglietto deve essere uguale o successiva alla data di vendita.

Per ogni  $b:\text{Biglietto}$ ,  $b.\text{data\_validità} \geq b.\text{data\_vendita}$ .

Non esiste  $b:\text{Biglietto}$  tale che  $b.\text{data\_validità} < b.\text{data\_vendita}$ .

#### Classe Biglietto\_standard

[V.Biglietto\_standard.prezzoCalcolato]

Il prezzo di un biglietto standard corrisponde al prezzo base della tariffa selezionata.

Per ogni  $b:\text{Biglietto\_standard}$ ,  $b.\text{prezzo} = b.\text{tariffa.prezzo\_base}$ .

#### Classe Biglietto\_ext\_acc

[V.Biglietto\_ext\_acc.prezzoCalcolato]

Il prezzo di un biglietto extended access è la somma del prezzo base della tariffa selezionata e dei prezzi delle esposizioni temporanee scelte.

Per ogni  $b:\text{Biglietto\_ext\_acc}$ ,  $b.\text{prezzo} = b.\text{tariffa.prezzo\_base} + e.\text{prezzo\_accesso}$

[V.Biglietto\_ext\_acc.coerenzaAccessi]

Ogni biglietto extended access dà sempre accesso a tutte le esposizioni permanenti.

Per ogni  $b:\text{Biglietto\_ext\_acc}$ ,  $b.\text{prezzo} = b.\text{tariffa.prezzo\_base}$

[O.Biglietto\_ext\_acc.calcola\_prezzo]

$\text{calcola\_prezzo}(\text{prezz\_acc: Prezzo}, \text{prezz\_base: Prezzo}): \text{Reale}$

Precondizione:

$\text{prezz\_acc} \geq 0 \wedge \text{prezz\_base} \geq 0$

Postcondizione:

$\text{risultato} = \text{prezz\_base} + \text{prezz\_acc}$  risultato  $\geq 0$

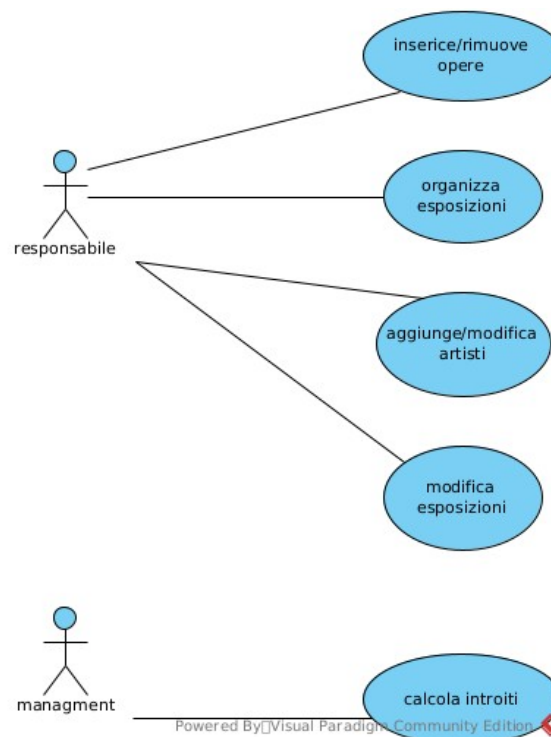
#### Classe Visitatore

[V.Visitatore.bigliettoSingoloPerData]

Un visitatore non può possedere più di un biglietto valido per la stessa data.

---

#### Diagramma Use-Case



## Specifica di UseCase:

### Specifica dello use case Inserisce/rimuove opere

inserisciOpera(o: Opera):

Precondizioni:

- o non è già presente nel sistema

Postcondizioni:

- o è aggiunta all'insieme delle opere gestite dal museo

rimuoviOpera(o: Opera):

Precondizioni:

- o è presente nel sistema

Postcondizioni:

- o è rimossa dall'insieme delle opere gestite dal museo

### Specifica dello use case Organizza esposizioni

organizzaEsposizione(O: opere):

Precondizioni:

- O sono già nel sistema
- O non sono presenti in altre esposizioni attualmente attive

Postcondizioni:

- O vengono aggiunte alle opere dell'Esposizione

### Specifica dello use case Aggiunge/modifica artisti

aggiungeArtista(a: Artista):

Precondizioni:

- a non è già presente nel sistema

Postcondizioni:

- a viene aggiunto all'insieme di artisti del museo

modificaArtista(a: Artista):

Precondizioni:

- a è già nel sistema

Postcondizioni:

- a viene modificato e sovrascritto nel sistema

### Specifica dello use case Modifica esposizioni

modificaEsposizione(e:Esposizione):

Precondizioni:

- e è già nel sistema

Postcondizioni:

- e viene modificato e sovrascritto nel sistema

### Specifica dello use case Registra vendite biglietti

registraVendite(B: biglietti, d: date):

Precondizioni:

- B sono stati venduti in d
- B hanno tipologia comune

Postcondizioni:

- le vendite vengono registrate nel sistema

Specifica dello use case Ottiene statistiche

calcolaIntroiti(e: EsposizioneTemporanea, t: timedelta): reale

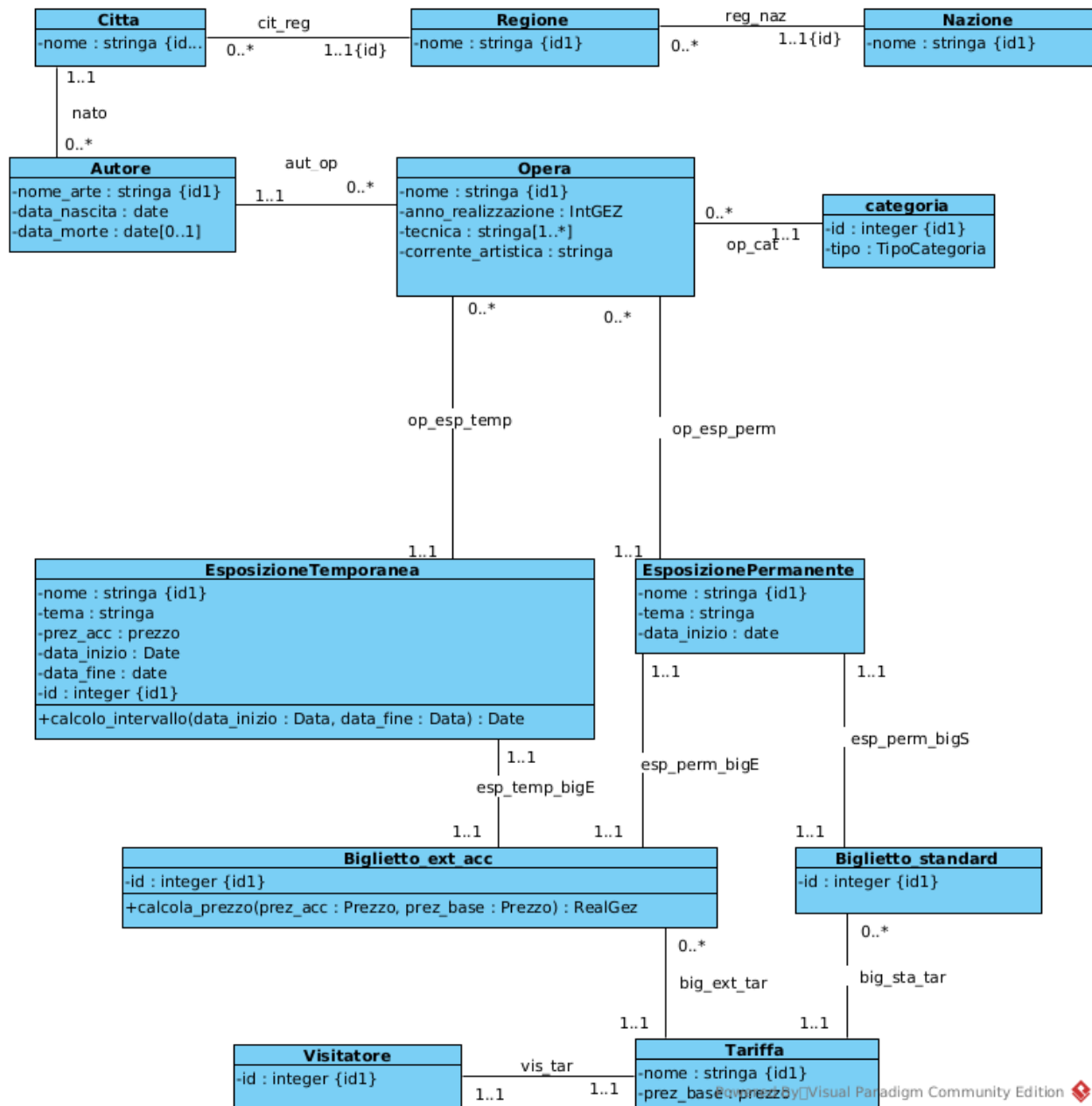
Precondizioni:

- e è un'esposizione temporanea registrata nel sistema
- t.inizio <= t.fine

Postcondizioni:

- result = somma dei prezzi di tutti i biglietti Extended Access venduti per e nel periodo

## Ristrutturazione UML per SQL



## Creazione di Domini e Tabelle – SQL

```
CREATE DOMAIN stringa as varchar (255);
CREATE DOMAIN IntGEZ as integer
check (value >= 0);
CREATE DOMAIN RealGEZ as real
check (value >= 0);
CREATE DOMAIN Prezzo as real
check (value >= 0);
```

```
CREATE TYPE TipoCategoria as ENUM
('Dipinto', 'Scultura', 'Mosaico', 'Manoscritto');
```

```
CREATE TABLE nazione (
nome stringa PRIMARY KEY
);
```

```
CREATE TABLE regione (
nome stringa NOT NULL,
nazione stringa NOT NULL,
PRIMARY KEY (nome, nazione),
FOREIGN KEY (nazione) REFERENCES nazione(nome)
);
```

```
CREATE TABLE citta (
nome stringa NOT NULL,
regione stringa NOT NULL,
nazione stringa NOT NULL,
PRIMARY KEY (nome, regione, nazione),
FOREIGN KEY (regione, nazione) REFERENCES regione(nome, nazione)
);
```

```
CREATE TABLE autore(
nome_arte stringa PRIMARY KEY,
data_nascita date NOT NULL,
data_morte date,
citta stringa NOT NULL,
regione stringa NOT NULL,
nazione stringa NOT NULL,
FOREIGN KEY (citta, regione, nazione) REFERENCES citta(nome, regione, nazione)
);
```

```
CREATE TABLE categoria (
id integer PRIMARY KEY,
tipo tipoCategoria NOT NULL
);
```

```
CREATE TABLE esposizionePermanente(
nome stringa PRIMARY KEY,
tema stringa NOT NULL,
data_inizio date NOT NULL
```

```
);
```

```
CREATE TABLE esposizioneTemporanea (  
  nome stringa PRIMARY KEY,  
  tema stringa NOT NULL,  
  prez_acc Prezzo NOT NULL,  
  data_inizio date NOT NULL,  
  data_fine date NOT NULL  
);
```

```
CREATE TABLE opera (  
  nome stringa PRIMARY KEY,  
  anno_realizzazione IntGEZ NOT NULL,  
  tecnica stringa NOT NULL,  
  corrente_artistica stringa NOT NULL,  
  categoria integer NOT NULL,  
  autore stringa NOT NULL,  
  esposizionePermanente stringa,  
  esposizioneTemporanea stringa,  
  FOREIGN KEY (categoria) REFERENCES categoria(id),  
  FOREIGN KEY (autore) REFERENCES autore(nome_arte),  
  FOREIGN KEY (esposizionePermanente) REFERENCES esposizionePermanente(nome),  
  FOREIGN KEY (esposizioneTemporanea) REFERENCES esposizioneTemporanea(nome),  
  CHECK (  
    (esposizionePermanente IS NOT NULL AND esposizioneTemporanea IS NULL)  
    OR (esposizionePermanente IS NULL AND esposizioneTemporanea IS NOT NULL)  
    OR (esposizionePermanente IS NULL AND esposizioneTemporanea IS NULL)  
  )  
);
```

```
CREATE TABLE visitatore(  
  id integer PRIMARY KEY  
);
```

```
CREATE TABLE tariffa(  
  nome stringa PRIMARY KEY,  
  prez_base prezzo NOT NULL  
);
```

```
CREATE TABLE biglietto_standard(  
  id integer PRIMARY KEY,  
  visitatore integer UNIQUE NOT NULL,  
  tariffa stringa NOT NULL,  
  esposizionePermanente stringa NOT NULL,  
  FOREIGN KEY (visitatore) REFERENCES visitatore(id),  
  FOREIGN KEY (tariffa) REFERENCES tariffa(nome),  
  FOREIGN KEY (esposizionePermanente) REFERENCES esposizionePermanente(nome)  
);
```

```
CREATE TABLE biglietto_ext_acc(  
  id integer PRIMARY KEY,  
  visitatore integer UNIQUE NOT NULL,
```



```

tariffa stringa NOT NULL,
esposizionePermanente stringa NOT NULL,
esposizioneTemporanea stringa NOT NULL,
FOREIGN KEY (visitatore) REFERENCES visitatore(id),
FOREIGN KEY (tariffa) REFERENCES tariffa(nome),
FOREIGN KEY (esposizionePermanente) REFERENCES esposizionePermanente(nome),
FOREIGN KEY (esposizioneTemporanea) REFERENCES esposizioneTemporanea(nome)
);

```

```
postgres=# \c myprecious
```

```
You are now connected to database "myprecious" as user "postgres".
```

```
myprecious=# \d
```

List of relations

Schema	Name	Type	Owner
public	autore	table	postgres
public	biglietto_ext_acc	table	postgres
public	biglietto_standard	table	postgres
public	categoria	table	postgres
public	citta	table	postgres
public	esposizionepermanente	table	postgres
public	esposizioneetemporanea	table	postgres
public	nazione	table	postgres
public	opera	table	postgres
public	regione	table	postgres
public	tariffa	table	postgres
public	visitatore	table	postgres

(12 rows)