**SCHELETRO**

L’entità OPERA contiene le informazioni relative alle opere che sono esposte nel museo o selezionate per l’allestimento di una mostra.

L’entità FORNITORE contiene le informazioni relative alla persona che vende o dona un’opera al museo.

L’entità EVENTO contiene le informazioni relative agli eventi che avvengono all’interno del museo.

L’entità GRUPPO contiene le informazioni relative ai gruppi di persone che partecipano ad un evento

L’entità VISITATORE contiene le informazioni relative alle persone che che intendono avere una relazione con il museo.

L’entità ARTICOLO contiene le informazioni relative ai prodotti in vendita online e al negozio stesso.

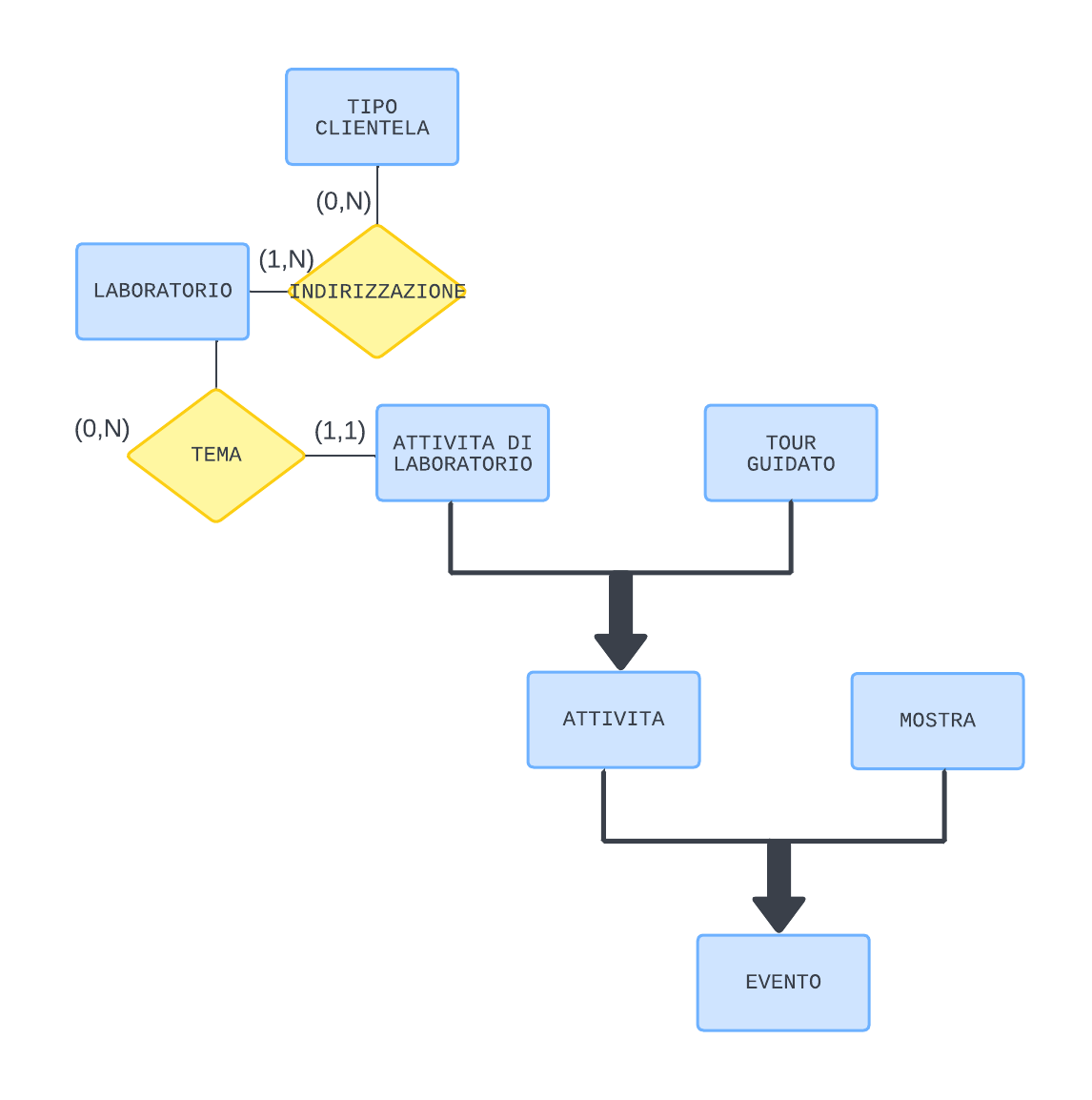
**EVENTO**

Un gruppo di visitatori può partecipare ai seguenti tipi di eventi:

* **Mostra**
* **Tour (guidato)**
* **Attività di laboratorio**

È necessario differenziarli per i seguenti motivi:

* Un gruppo può partecipare ad una mostra, ma non può prenotarsi per parteciparvi ed inoltre non è prevista l’illustrazione da parte di una guida
* Un gruppo può partecipare ad un’attività di laboratorio o ad un tour guidato con la possibilità di effettuare una prenotazione in anticipo essendo affiancato da una guida
* Per svolgere un’attività di laboratorio, è necessaria la scelta di un tema a cui corrisponde una durata ed un prezzo.
* Ogni tema per le attività di laboratorio è indirizzato ad uno o più tipologie di clientela. (vedi tabella)



INIZIO

**STRATEGIA**

Termitata l’intervista, si è proceduto con l’organizzazione ed il chiarimento delle specifiche dei requisiti, ricavandone una descrizione in linguaggio naturale.

Fatto ciò, oltre al’individuazione delle entità più chiavi di questo sistema, è stato naturale ricavare le relazioni basiche tra loro. Questo processo ha dato vita allo SCHELETRO di questa base di dati.

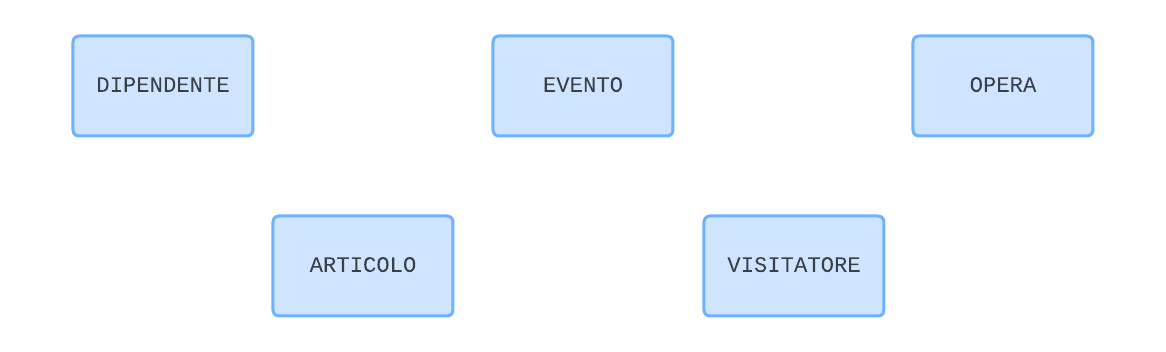
L’approccio utilizilizzato per lo sviluppo di questa struttura è stato quindi misto, sfruttando a pieno i vantaggi delle strategie **top-down** e **bottom-up**.

In tale progettazione è presente la metodologia **top-down,** **in quanto (poiché)** si è partit**i** da uno schema grezzo, coinvolgendo inizialmente solo le figure fondamentali, per poi completarlo e raffinarlo in ogni dettaglio. Inoltre, nella correzione dello schema, si sono intravisti dei problemi di progettazione simili a quelli documentati nei design pattern, perciò sono state scelte le soluzioni suggerite dagli stessi.

È stato invece sfruttato l’approccio **bottom-up** per integrare le varie “isole” individuate precedentemente, aggregandole facendo riferimento allo scheletro iniziale, al fine di ottere un diagramma concettuale che esprima al meglio la soluzione ricercata.

**IDENTIFICAZIONE DELLE ENTITÀ E RELAZIONI FONDAMENTALI**

Come anticipato nel paragrafo precedente, analizzando i requisiti una volta riordinati, si è riusciti ad ottenere una visione generale del problema, identificando le entità protagoniste dei seguenti macroblocchi principali: DIPENDENTE, OPERA, EVENTO, VISITATORE, ARTICOLO.



Con DIPENDENTE, si intende il componente che descrive i vari tipi dipendente avente, seppur non necessariamente, un ruolo all’interno della struttura museale.

Con OPERA, si intende il macroblocco contenente la descrizione di tutte le opere che il museo intende far vedere e toccare.

Con EVENTO, si intende il macroblocco contenente i vari tipi evento, tenendo conto del flusso di partecipanti.

Con VISITATORE, si intende il macroblocco contenente i vari tipi di visitatori che interagiscono con il museo.

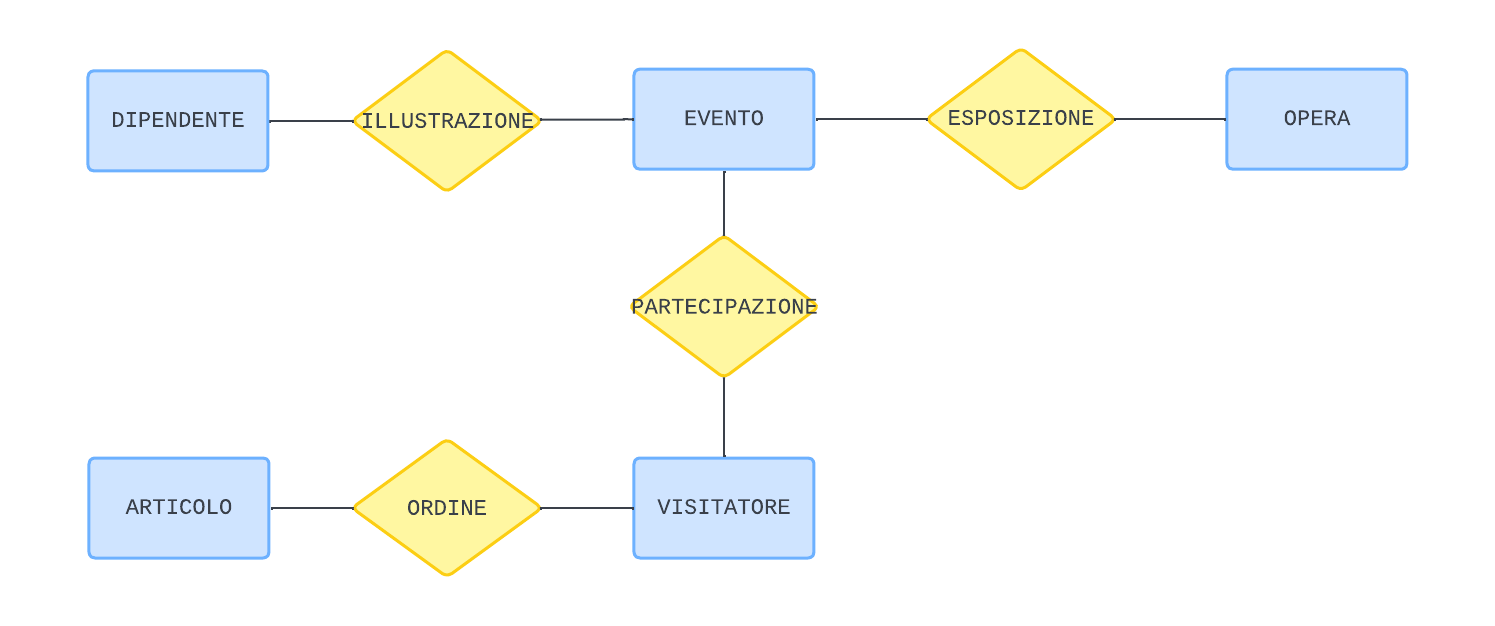
Con ARTICOLO, si intende il macroblocco riguardante la gestione dei prodotti in vendita al negozio, sia fisico che online.

**SCHELETRO**

Dalle entità individuate nel paragrafo precedente, è stato possibile ottenere uno schema iniziale, fungente da linea guida per il resto della progettazione di questo schema concettuale.

Si nota come i VISITATORI, mediante la partecipazione ad EVENTI, interagiscono con il museo, che, cercando di migliorare tale esperienza, organizza per queste occasioni un’esposizione di OPERE.

Inoltre, gli EVENTI, vengono illustrati dai DIPENDENTI, per accompagnare i visitatori. Quest’ultimi possono acquistare ARTICOLI mediante l’effettuazione di un ordine.

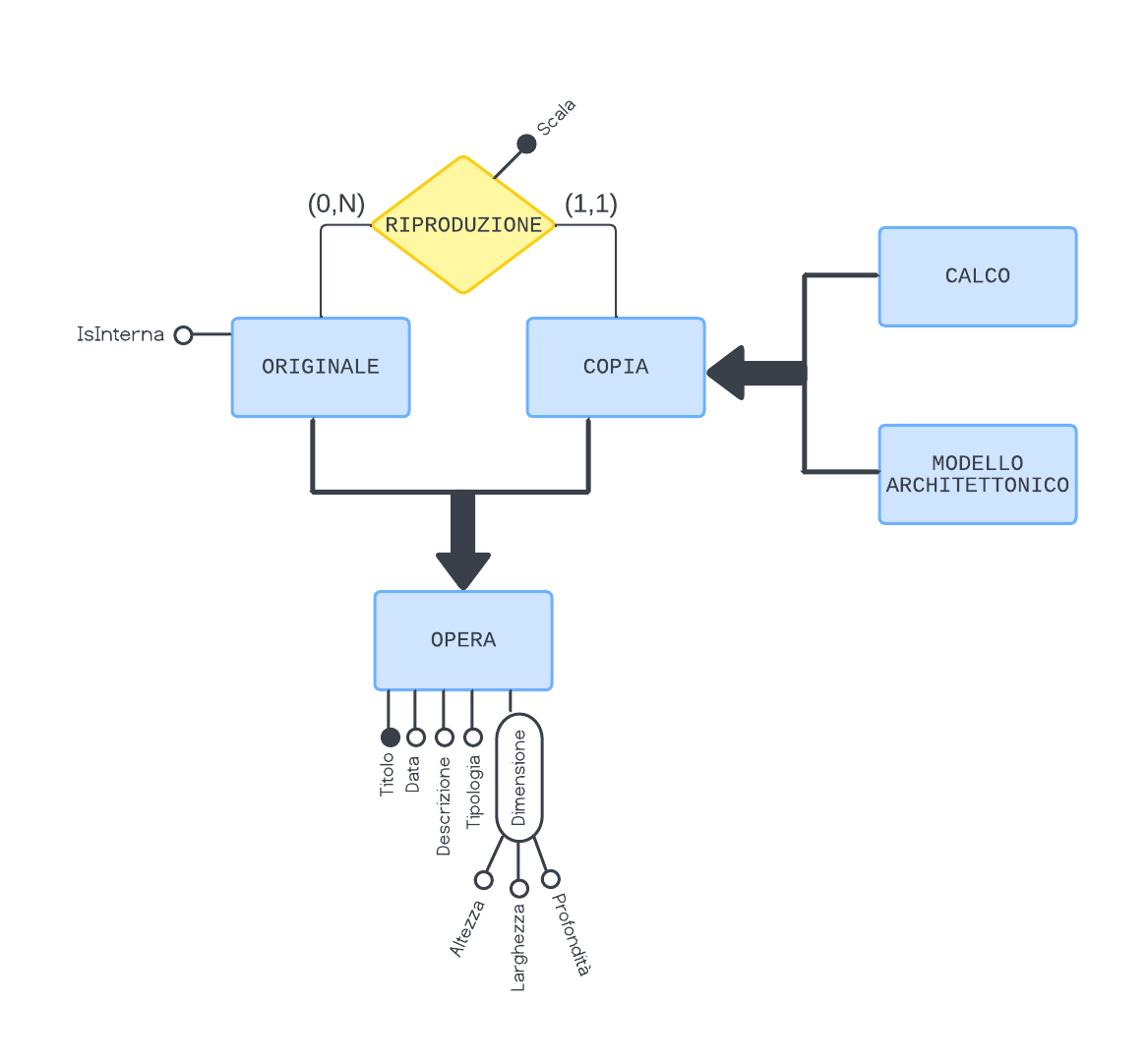


**2.4 SVILUPPO COMPONENTI SCHELETRO**

Una volta definito lo scheletro, si è deciso di proseguire la progettazione dello schema con un approccio **top-down**, raffinando i componenti presenti singolarmente. Durante lo sviluppo, sono stati introdotti anche nuovi concetti necessari per soddisfare alcuni requisiti che non erano stati trattati nello schema iniziale (fase **bottom-up**).

**2.4.1 OPERA**

L’entità OPERA è la principale di questo schema. Prima di analizzare le sue relazioni, è necessario comprendere cosa rappresenta l’entità stessa.

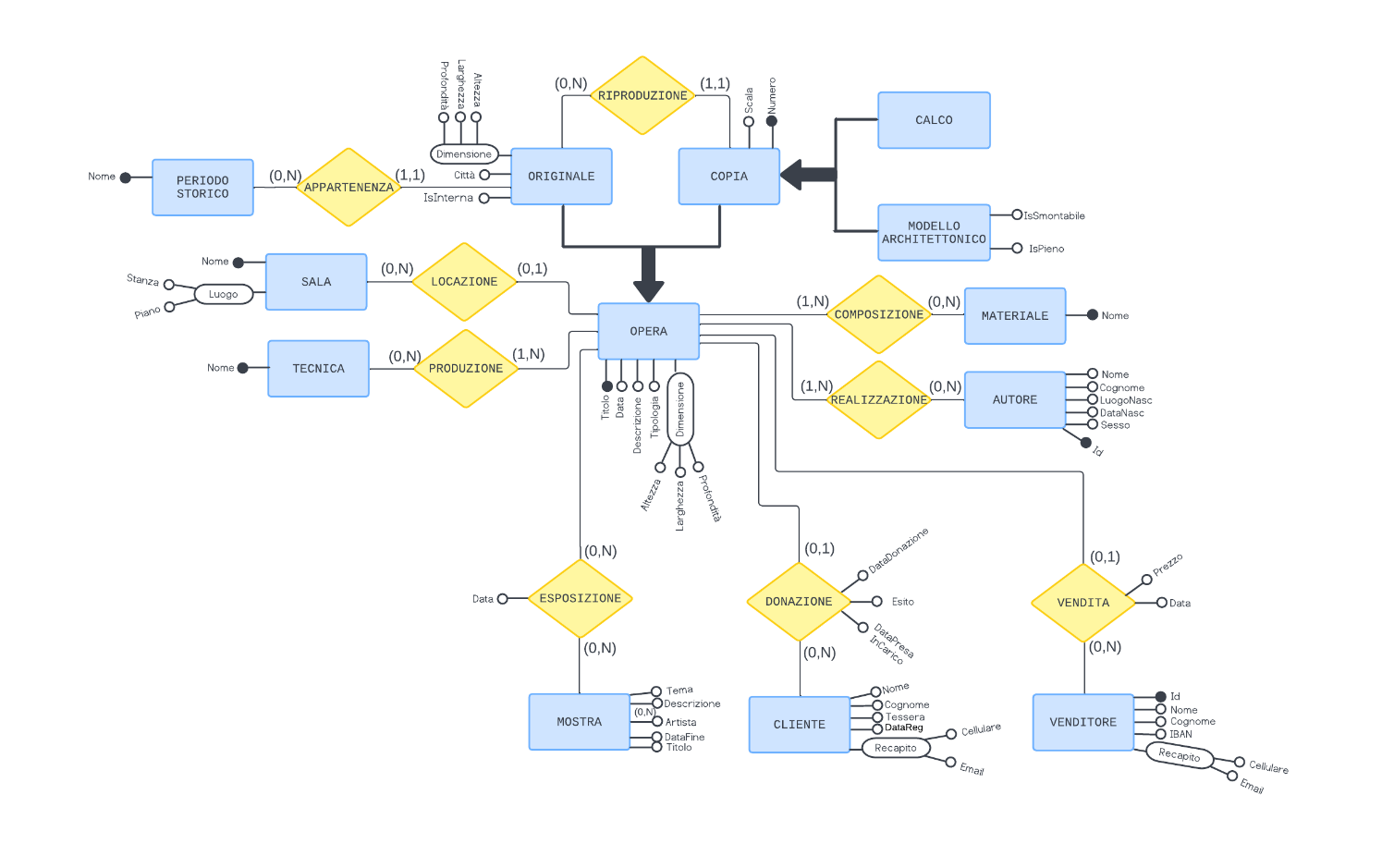


Alcune opere, appena realizzate, vengono direttamente esposte nel museo, altre invece sono una riproduzione di un’opera realizzata nel passato, in cui può variare la dimensione (progettazione in scala) ed il materiale utilizzato.

Nasce la relazione “Riproduzione” perché una copia può rappresentare più opere originali che possono essere rappresentate al massimo da una copia.

Nasce la relazione “Riproduzione” uno a molti che associa “Copia” con cardinalità (1,1) all’entità “Originale” con cardinalità (0,N). Infatti, un’opera originale, può avere più copie esposte nel museo, ma non è necessario che ne abbia una. Se tale opera non è contenuta nel museo, ovvero si possiedono al massimo delle riproduzioni, l’attributo “IsInterna” viene impostato a “false”.

Dall’entità “Copia”, si ottiene una gerarchia totale in quanto è un calco o un modello architettonico.



Il museo può ottenere un’opera da un donatore o da un venditore, considerando il primo come un cliente. La relazione “Donazione” viene considerata differente dalla relazione “Vendita” in quanto essa deve essere presa in carico con un eventuale rifiuto o accettazione, mentre nella seconda questo non è possibile in quanto il museo stesso ne è pagante.

Con l’entità “Autore”, si intende memorizzare chi ha realizzato un’opera, sia che si tratti dell’originale (compresi eventuali personaggi del passato) che della copia. Nasce la relazione “Realizzazione” molti a molti, in quanto più persone possono lavorare alla stessa opera, contribuendo contemporaneamente alla produzione di diverse opere.

Tale produzione può avvenire da tali autori mediante una o più tecniche di lavoro, ottenendo una composizione formata da uno o più materiali. Nascono quindi altre due relazioni molti a molti sotto il nome di “Produzione” e “Composizione”, dove la prima è un’associazione con l’entità “Tecnica” e la seconda è una’associazione con l’entità “Materiale”.

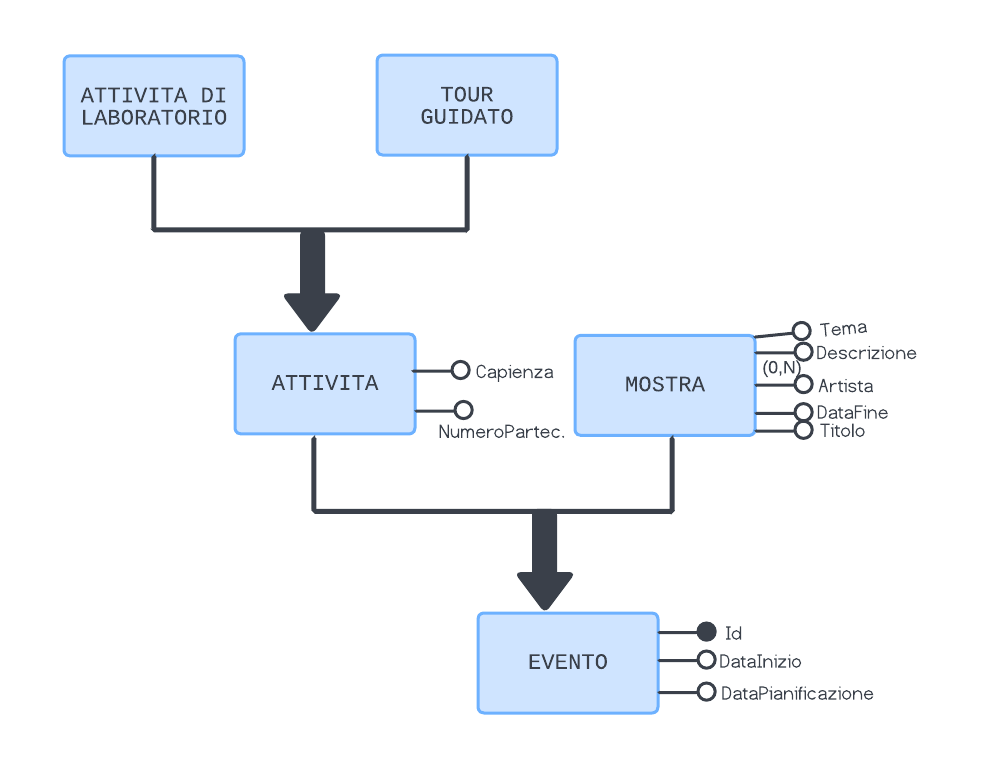
Nasce inoltre la relazione “Locazione” uno a molti con l’entità “Sala” per memorizzare la posizione di un’opera, che può condividere lo stesso spazio.

La relazione “Appartenenza” uno a molti, associa l’entità “Periodo Storico” con “Originale” e non con l’entità “Opera” in quanto la copia non è altro che la riproduzione dell’originale.

Le entità “Periodo Storico” e “Originale” sono associate dalla relazione “Appartenenza” uno a molti, in quanto più modelli originali possono essere stati realizzati nello stesso periodo, ma possono appartenere solo ad uno.

**2.4.2 EVENTO**

In questa base di dati, svolge un ruolo principale l’entità “EVENTO”. Come nel paragrafo **NUMERO PARAGRAFO OPERA TODO**, prima di analizzare le sue relazioni, è necessario comprendere cosa rappresenta tale entità.

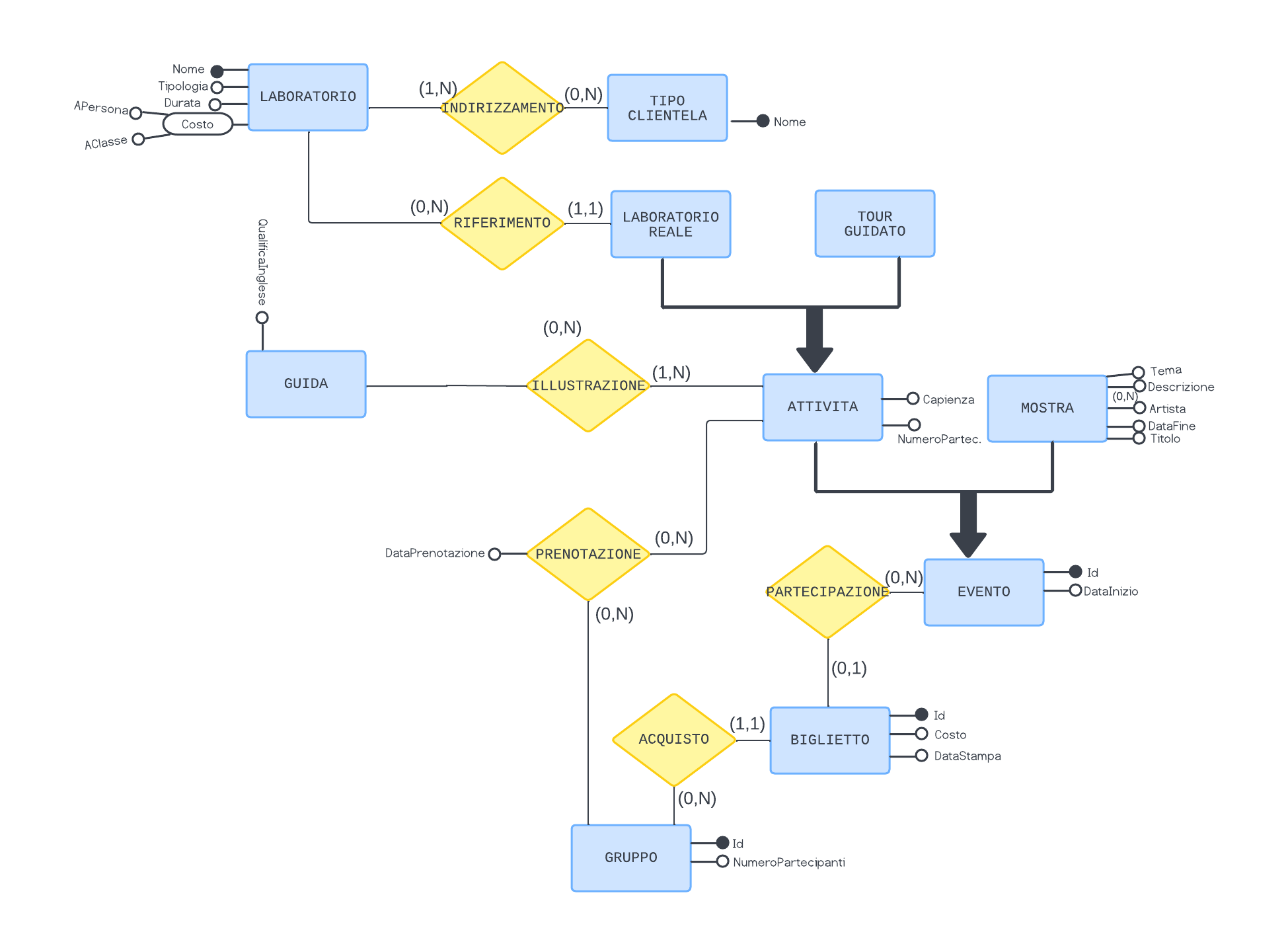


**VALERIO GUARDA QUESTA RIGA SOTTO QUESTO COMMENTO**

Come possibile notare dall’analisi dei requisiti **NUMERO PARAGRAFO ANALISI REQUISITI TODO,** un gruppo di visitatori può interagire con il museo partecipando ai seguenti tipi di eventi:

* **Mostra**
* **Tour (guidato)**
* **Attività di laboratorio**

Tali gerarchie non sono altro che la conseguenza di un’accomunazione di attributi e vincoli *(di cosa TODO?)*. Al fine di comprendere al meglio quest’ultimi, che portano alla necessità di differenziare tali eventi in questo schema, si intende procedere con l’analisi delle relazioni che coinvolgono queste entità.



Un gruppo può prenotarsi per un tour guidato o per un’attività di laboratorio nei quali possono essere inseriti più gruppi. Nasce quindi la relazione “Prenotazione” molti a molti tra l’entità “Gruppo” e “Attività”.

Un’attività può essere illustrata da una o più guide, viene dunque data forma alla relazione “Illustrazione” molti molti a molti con cardinalità (1,N) nell’associazione con “Attività” in quanto è SEMPRE previsto l’accompagnamento di una guida. Per essere coerenti con questa decisione, si è deciso di non registrare come attività, l’ingresso di un gruppo al museo senza una guida, per cui, allo stesso tempo, non ne è nemmeno prevista una prenotazione.

Per tenere in considerazione l’effettivo flusso di persone al museo, viene utilizzata l’entità “Biglietto”. Attraverso tale entità si intende l’accesso ad un evento o l’ingresso libero al museo. Al fine di registrare quest’ultimo, l’entità “Biglietto” è associata alla relazione “Partecipazione” con cardinalità (0,1).

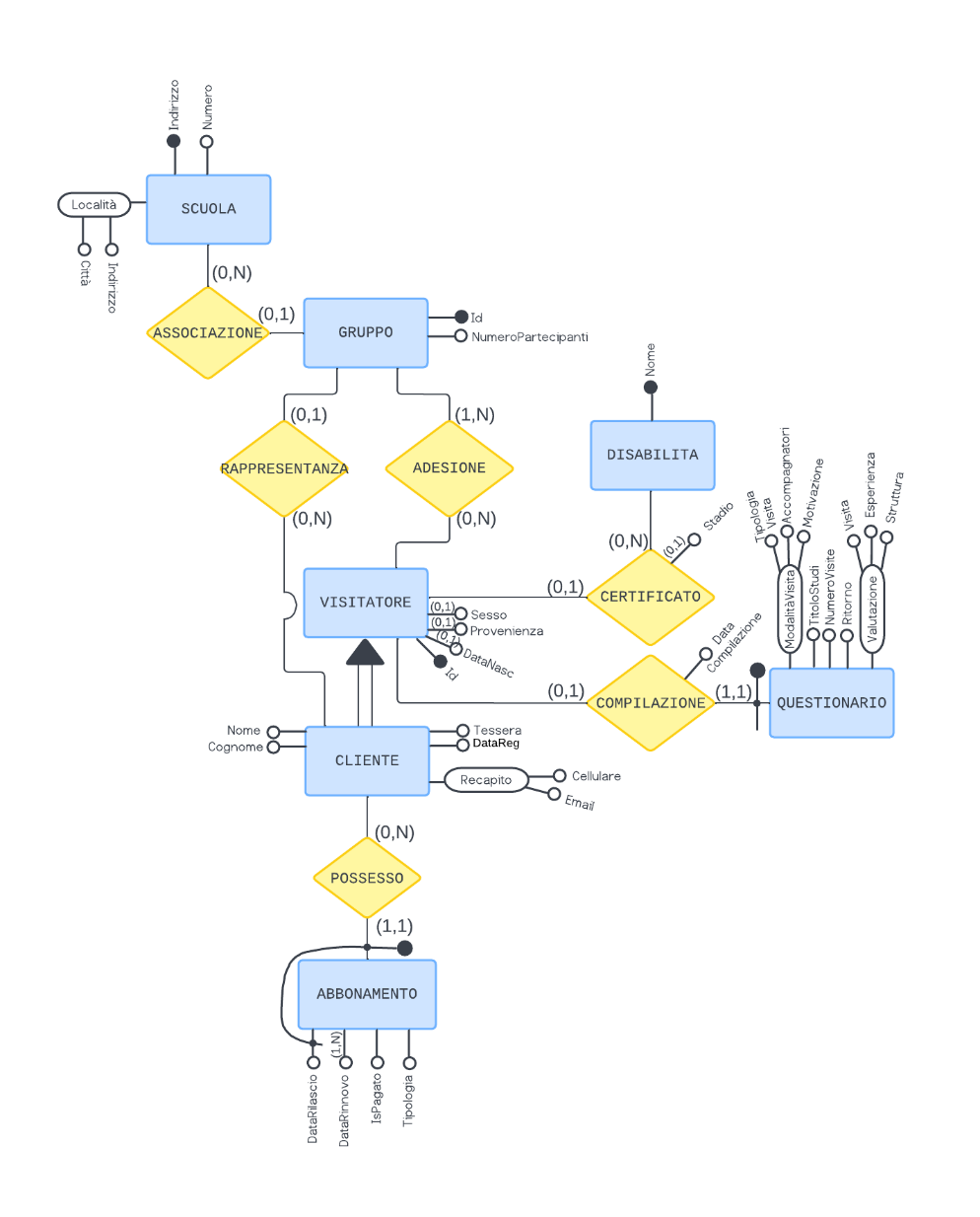
L’acquisto di un biglietto può essere relativo ad uno ed un gruppo soltanto, formando un’associazione con cardinalità (1,1) con la relazione “Acquisto”. Nel caso in cui un gruppo volesse partecipare a più di un evento nello stesso giorno, avviene l’acquisto di più biglietti. Si ottiene dunque un’associazione tra le entità “Gruppo” e “Acquisto” con cardinalità (0,N). Se lo stesso gruppo si presenta in giorni differenti, non viene utilizzato lo stesso ma ne viene creato uno nuovo.

La relazione “Riferimento” tra le entità “Laboratorio Reale” e “Laboratorio”, sono frutto di un **del design pattern “istanza di”.** Con “Laboratorio Reale” si intende l’avvenimento (o pianificazione) di un evento, mentre con “Laboratorio” si intende il tipo di attività che viene svolta, quindi il tema. È possibile visulizzare l’elenco dei temi nella foto **XXXXX**. Tra “Laboratorio Reale” e tale relazione, c’è una cardinalità (1,1) in quanto deve essere scelto obbligatoriamente un tema. Ognuno di questi è indirizzato uno o più tipi di clientela, nasce quindi la relazione “Indirizzamento” molti a molti. Per comprendere al meglio questa relazione si rimanda alla foto **XXXXX.**

Come specificato nel paragrafo precedente, un gruppo può partecipare ad un qualsiasi evento mediante un BIGLIETTO che viene “acquistato” all’entrata del museo, inoltre può prenotarsi per un’attività.

Tale gruppo, in caso di un eventuale prenotazione, deve essere rappresentato da una persona che rilascia le informazioni necessarie nel caso in cui il museo avesse la necessità di contattarlo. Inoltre, per ogni gruppo, si tiene traccia dei VISITATORI che vi appartengono in quanto ad ognuno di essi viene applicata una tariffa in base ad un’eventuale DISABILITÀ o ABBONAMENTO.

**GRUPPO**

Come già specificato più volte, l’ingresso al museo viene considerato a gruppi di persone e non ad individui. Occorre quindi un’analisi dettagliata del ruolo dell’entità “Gruppo” all’interno di questa base di dati.

Tale entità, viene associata a “Visitatore” tramite la relazione “Adesione” molti a molti con cardinalità (1,N). Ciò comporta che un gruppo non può essere nullo e che un visitatore può appartenere a più gruppi differenti. Nel caso in cui quest’ultimo si presenti più volte al museo senza essere un cliente (si rimanda al dizionario dei dati per la definizione dettagliata di “visitatore” e “cliente”), ne viene creata una nuova occorrenza, altrimenti la stessa persona viene inserita coma aderente ad un nuovo gruppo.

Dalla possibile necessità di contattare il responsabile di un gruppo, nasce la relazione “Rappresentanza” con “Cliente”. La relazione non avviene con l’entità “Visitatore” in quanto per questo concetto occorre una figura reperibile, di cui si vogliono conoscere nome, cognome ed un recapito, quindi una persona registrata. La stessa può rappresentare gruppi diversi nel tempo, che possono essere rappresentati da un solo cliente, si parla quindi di una relazione uno a molti.

Per memorizzare i gruppi scolastici che interagiscono con il museo, nasce la relazione “Associazione” uno a molti con l’entità “Scuola”, in quanto una scuola può far visita al museo più volte (con gruppi diversi), ma ad un gruppo è associata solo una scuola.

Oltre a delle domande a cui il visitatore può avvalersi della facoltà di non rispondere, quali gli attributi con cardinalità (0,1) dell’entità “Visitatore”, ne vengono identificate altre con “Questionario”, associato alla prima entità tramite la relazione “Compilazione”. Il questionario è necessariamente associato ad un visitatore, quindi si associa alla relazione con cardinalità (1,1), ma allo stesso tempo il visitatore può rifiutare l’invito alla compilazione, quindi si associa alla relazione con cardinalità (0,1).

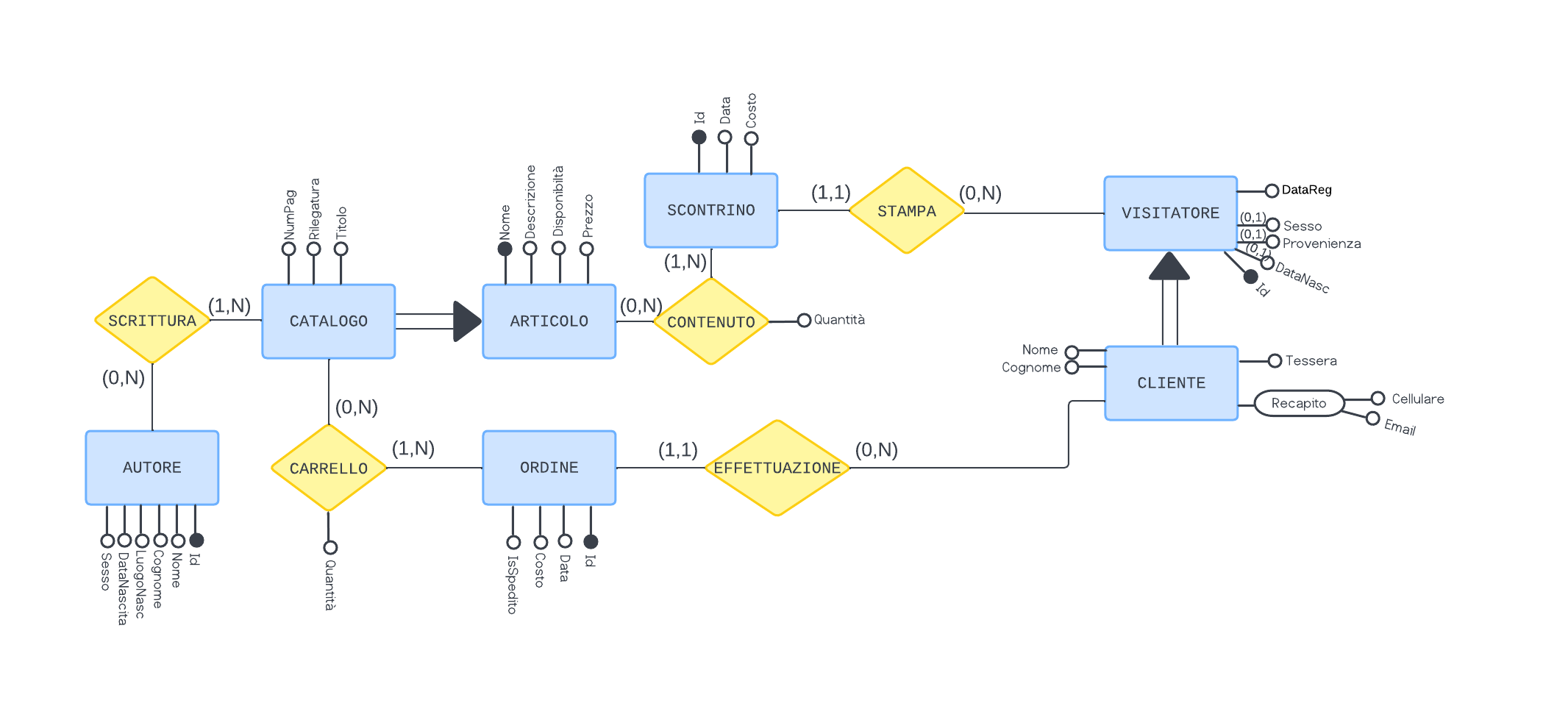
Un altro dato importante da conoscere riguardo a “Visitatore”, è una sua eventuale “Disabilità”, tale entità si associa alla prima tramite la relazione “Certificato” uno a molti con cardinalità (0,N), mentre dal lato di “Visitatore” si ha una cardinalità (0,1) in quanto è un concetto che riguarda un ristretto gruppo di persone.

Inoltre il visitatore può scegliere di effettuare un abbonamento, in tal caso deve essere registrato, quindi si parla di un “Cliente”, che si associa alla relazione “Possesso” con cardinalità (0,N) in quanto non è obbligatorio, mentre si ha cardinalità (1,1) nell’associazione tra la relazione e l’entità “Abbonamento” in quanto deve possedere un titolare.

Le ultime due relazioni, ovvero “Certificato” e “Possesso”, oltre ad aggiungere informazioni sul visitatore o sul cliente, **contribuiscono a determinare** il prezzo del biglietto, che può variare a seconda di una disabilità o di un abbonamento.

**ARTICOLO**

Il negozio del museo va osservato su due fronti differenti: lo store fisico e lo store online. Essi sono legati fra loro, ma non sono la stessa cosa. Segue il diagramma ER dedicato alla rappresentazione di questo concetto.



Per poter acquistare degli articoli al negozio fisico, ovviamente non è necessario che il pagante sia un utente registrato, mentre, per il negozio online, è necessaria un’iscrizione ai fini del pagamento online e della consegna. Quindi, da tali ragionamenti, si deriva che il museo si interfaccia con un visitatore al negozio fisico e con un cliente al negozio online.

Analizzando il primo caso, partendo dalla figura pagante, ovvero “Visitatore”, si ricava una relazione “Stampa” uno a molti con l’entità “Scontrino”. Una persona può acquistare più oggetti in momenti differenti, ricevendo quindi più scontrini, ma senza essere obbligato a comprare qualcosa. Ciò comporta una cardinalità (0,N) dal lato del visitatore, mentre, poiché “Scontrino” deve essere relativo ad un acquisto effettuato da qualcuno, si associa a “Stampa” con cardinalità (1,1).

Lo stesso ragionamento è applicabile tra l’entità “Cliente” ed “Ordine”, legate dalla relazione “Effettuazione” uno a molti. A differenza di “Scontrino”, in questo caso c’è un pacco da spedire, quindi si vuole sapere se l’ordine è stato spedito o è in fase di preparazione tramite l’attributo “IsSpedito”.

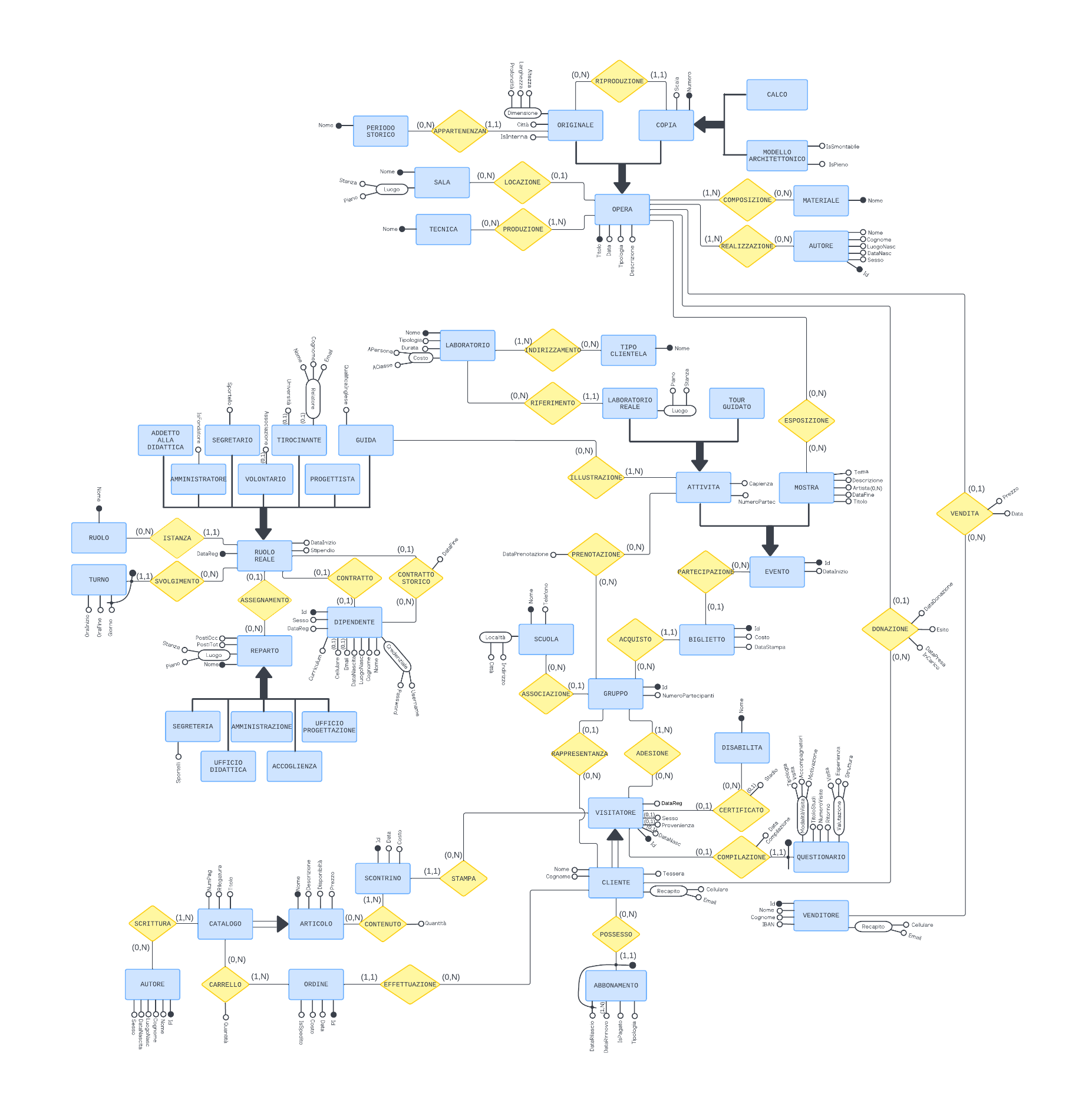
Tra “Scontrino” e “Articolo”, nasce la relazione “Contenuto” molti a molti. La prima si associa alla relazione con cardinalità (1,N), in quanto si possono acquistare più cose contemporaneamente, ma si suppone che uno scontrino non possa essere vuoto. Invece, la seconda si associa alla relazione con cardinalità (0,N) in quanto lo stesso prodotto può essere inserito più volte in scontrini differenti e, nel peggiore dei casi, non essere acquistato da nessuno (di persona).

Non tutti gli articoli presenti al negozio fisico sono presenti nel negozio online, poiché in quest’ultimo vengono venduti solo i cataloghi. Nasce quindi una gerarchia (un sottoinsieme nello specifico) in quanto un catalogo è un articolo sul mercato, ma un articolo non è necessariamente un catalogo poiché ci sono altri prodotti in vendita.

Quindi, per memorizzare il contenuto di un ordine, l’entità “Ordine” non si associa all’entità “Articolo”, ma a “Catalogo”, tramite la relazione “Carrello” molti a molti. Il ragionamento applicato a questa relazione è il medesimo applicato a “Contenuto”.

Infine, tra l’entità “Catalogo” e l’entità “Autore”, nasce la relazione “Scrittura” molti a molti. La prima si associa alla relazione con cardinalità (1,N), in quanto un catalogo può essere scritto da uno o più autori, mentre la seconda si associa alla relazione con cardinalità (0,N) in quanto lo stesso autore può aver scritto più cataloghi nella sua vita.

**DIAGRAMMA ER**

Unendo i macroblocchi descritti precedentemente, si è ottenuta la seguente mappa concettuale corretta, completa e leggibile.

**Dipendenti** (Id, Nome, Cognome, DataNasc, LuogoNasc, Sesso, Email, Cellulare, Curriculum, DataRegistrazione)

**Credenziali** (dipendente, Username, Password)

**Reparti** (nome, piano, stanza, postiTot, PostiOcc)

**RuoliReali** (dipendente, ruolo, dataFine, dataInizio, stipendio, isStorico)

**Ruoli** (nome)

**Turni** (dipendente, giorno, oraInizio, oraFine)

**Volontari** (dipendente, Associazione)

**Tirocinanti** (dipendente, relatore)

**Relatori** (id, nome, cognome, università, email)

**Guide** (dipendente, qualificaInglese)

**Scontrini** (id, data, costo, visitatore)

**Articoli** (nome, descrizione, disponibilità, prezzo)

**Contenuti** (Articolo, scontrino, quantità)

**Cataloghi** (nome, numeroPag, rilegatura, titolo)

**Autori** (Id, Nome, Cognome, DataNasc, LuogoNasc, Sesso)

**Scrittura** (autore, catalogo)

**Ordini** (Id, data, costo, IsSpedito, cliente)

**Carrello** (Ordine, catalogo, quantità)

**Abbonamenti** (cliente, dataRilascio, dataScadenza, costo, tipologia)

**Clienti** (id, nome, cognome, tessera, cellulare, email, dataFine, dataInizio, stipendio, isStorico)

**Visitatore** (id, tariffa)

**Questionario** (visitatore,

infVisitatore (visitatore, sesso, provenienza, dataNascita)

Disabilità (nome)

Certificati (disabilità, visitatore, stadio)

Gruppi (id, numPartecipanti)

Adesioni gruppo visitatore

Rappresentanze (gruppo, cliente)

Scuole (nome, città, indirizzo, telefono)

Associazioni (gruppo, scuola)

Biglietti (id, gruppo, evento, costo, dataStampa)

Eventi (id, oraInizio)

Mostre (evento,

Esposizioni (mostra, opera)

Attività (evento, IsTourGuidato, capienza, numeroPartecipanti)

Prenotazioni (gruppo, attività, dataPrenotazione)

Illustrazioni (attività, guida)

LaboratorioReale (attività, laboratorio, piano, stanza)

Laboratorio (nome, tipologia, durata, costoPersona, costoClasse)

TipoClientela (nome)

Indirizzamenti (laboratorio, tipoClientela)

Venditori (id, nome, cognome, IBAN, cellulare, email)

Opere (titolo, data, tipologia, descrizione, sala)

Donazioni (opera, donatore, dataDonazione, dataPresaInCarico, esito)

Vendite (opera, venditore, prezzo, data)

Tecniche (nome)

Produzione (opera, tecnica)

Sale (nome, stanza, piano)

Materiali (nome)

Composizione (opera, materiale)

Artisti (id, nome, cognome, luogoNascita, dataNascita, sesso)

Realizzazioni (opera, artisti)

Originali (titolo, IsInterna, città, altezza, larghezza, profondità, periodoStorico, isInterna)

PeriodiStorici (nome)

Copie (titolo, numero, IsCalco, scala)

ModelliArchitettonici (Copia, IsSmontabile, IsPieno)

create table reparti(

nome varchar(20) primary key,

piano int,

stanza varchar(3),

posti\_tot int,

posti\_occ int,

check (posti\_occ <= posti\_tot)

);

create table ruoli(

nome varchar(25) primary key

);

Create table credenziali(

Dipendente integer not null references dipendenti(id) on update cascade on delete cascade,

username varchar(20) unique not null,

password varchar(20) not null,

unique(username, password),

primary key (dipendente)

);

create table dipendenti(

id integer auto\_increment primary key,

nome varchar(20) not null,

cognome varchar(20) not null,

data\_nasc date not null,

luogo\_nasc varchar(20) not null,

sesso character not null check (sesso = 'M' or sesso = 'F'),

email varchar(25) unique,

cellulare varchar(15) unique,

curriculum varchar(50),

data\_registrazione datetime default current\_timestamp,

unique(nome, cognome, data\_nasc, luogo\_nasc)

);

create table ruoli\_reali(

dipendente integer not null references dipendenti(id) on update cascade on delete cascade,

ruolo varchar(25) not null references ruoli(nome) on update cascade on delete set null,

data\_inizio date,

data\_fine date,

stipendio integer default 0,

is\_storico bit not null default 0,

reparto varchar(20) references reparti(nome) on update cascade on delete set null,

primary key(dipendente, data\_inizio)

);

create table volontari(

dipendente integer not null references ruoli\_reali(dipendente) on update cascade on delete cascade,

data\_inizio date not null references ruoli\_reali(data\_inizio) on update cascade on delete cascade,

associazione varchar(20),

primary key(dipendente, data\_inizio)

);

create table guide(

dipendente integer not null references ruoli\_reali(dipendente) on update cascade on delete cascade,

data\_inizio date not null references ruoli\_reali(data\_inizio) on update cascade on delete cascade,

qualifica\_inglese varchar(10),

primary key(dipendente, data\_inizio)

);

create table tirocinanti(

dipendente integer not null references ruoli\_reali(dipendente) on update cascade on delete cascade,

data\_inizio date not null references ruoli\_reali(data\_inizio) on update cascade on delete cascade,

primary key(dipendente, data\_inizio)

);

create table relatori(

id integer auto\_increment primary key,

nome varchar(20) not null,

cognome varchar(20) not null,

università varchar(30),

email varchar(25),

unique (nome, cognome, università)

);

create table turni(

dipendente integer not null references ruoli\_reali(dipendente) on update cascade on delete cascade,

data\_inizio date not null references ruoli\_reali(data\_inizio) on update cascade on delete cascade,

giorno enum('lunedi', 'martedi', 'mercoledi', 'giovedi', 'venerdi', 'sabato', 'domenica'),

ora\_inizio time,

ora\_fine time,

primary key(dipendente, data\_inizio, giorno)

);

create table sale(

nome varchar(25) primary key,

piano int,

stanza varchar(3)

);

create table tecniche(

nome varchar(25) primary key

);

create table produzioni(

opera integer not null references opere(id) on update cascade on delete cascade,

tecnica varchar(25) not null references tecniche(nome) on update cascade on delete cascade,

primary key (opera, tecnica)

);

create table materiali(

nome varchar(25) primary key

);

create table periodi\_storici(

nome varchar(25) primary key

);

create table composizioni(

opera integer not null references opere(id) on update cascade on delete cascade,

materiale varchar(25) not null references materiali(nome) on update cascade on delete cascade,

primary key (opera, materiale)

);

create table artisti(

id integer auto\_increment primary key,

nome varchar(20) not null,

cognome varchar(20) not null,

data\_nasc date not null,

luogo\_nasc varchar(20) not null,

sesso character not null check (sesso = 'M' or sesso = 'F'),

unique (nome, cognome, data\_nasc, luogo\_nasc)

);

create table realizzazioni(

opera integer not null references opere(id) on update cascade on delete cascade,

artista integer not null references artisti(id) on update cascade on delete no action,

primary key (opera, artista)

);

create table opere(

id integer auto\_increment primary key,

data date,

tipologia enum('Foglio', 'Quadro', 'Tela', 'Bassorilievo', 'Modello', 'Statua', 'Libro'),

descrizione varchar (500),

sala varchar(25) references sale(nome) on update cascade on delete

set null

);

create table originali(

id integer not null references opere(id) on update

cascade on delete cascade,

titolo varchar(30) not null unique,

is\_interna bit not null,

città varchar(20),

altezza varchar(8),

larghezza varchar(8),

profondità varchar(8),

periodo\_storico varchar(25) not null references periodi\_storici(nome) on update cascade on delete set null,

primary key(id)

);

Create table venditori(

id integer auto\_increment primary key,

nome varchar(20) not null,

cognome varchar(20) not null,

iban varchar(20),

cellulare varchar (20),

email varchar(25),

unique (nome, cognome, email)

);

create table vendite(

opera integer not null references opere(id) on update cascade on delete cascade,

venditore integer not null references venditori(id) on update cascade on delete set null,

primary key (opera, venditore)

);

create table donazioni(

opera integer not null references opere(id) on update cascade on delete cascade,

donatore integer not null references clienti(id) on update cascade on delete no action,

data\_donazione date,

data\_presa\_in\_carico date not null,

esito enum ('positivo', 'negativo'),

check (data\_donazione <= data\_presa\_in\_carico),

primary key (opera)

);

Create table gruppi(

Id integer primary key,

num\_partecipanti int not null,

check (num\_partecipanti > 0)

);

create table visitatori(

id integer primary key,

tariffa enum('intero', 'ridotto', 'gratuito') not null default 'intero'

);

create table informazioni\_visitatori(

visitatore integer not null references visitatori(id) on update cascade on delete cascade,

provenienza varchar(20),

sesso character check (sesso = 'M' or sesso = 'F'),

data\_nasc date,

primary key(visitatore)

);

create table disabilita (

nome varchar(20) primary key

);

Create table certificati(

visitatore integer not null references visitatori(id) on update cascade on delete cascade,

disabilita varchar(20) not null references disabilita(nome) on update cascade on delete cascade,

primary key (visitatore)

);

Create table adesioni(

gruppo integer not null references gruppi(id) on update cascade on delete cascade,

visitatore integer not null references visitatori(id) on update cascade on delete cascade,

primary key (gruppo, visitatore)

);

Create table rappresentanze(

gruppo integer not null references gruppi(id) on update cascade on delete cascade,

rappresentante integer not null references clienti(id) on update cascade on delete cascade,

primary key (gruppo)

);

create table scuole(

nome varchar(25) primary key,

città varchar(20),

indirizzo varchar(25),

telefono varchar(15)

);

create table associazioni(

gruppo integer not null references gruppi(id) on update cascade on delete cascade,

scuola varchar(25) not null references scuole(nome) on update cascade on delete no action,

primary key (gruppo)

);

Create table biglietti(

Id varchar(15) primary key,

gruppo integer not null references gruppi(id) on update cascade on delete set null,

evento integer references eventi(id) on update cascade on delete set null,

costo decimal(6,2) not null check(costo >= 0),

data\_stampa datetime not null default current\_timestamp

);

create table articoli(

nome varchar(30) primary key,

descrizione varchar (100),

disponibilita int not null default 0,

prezzo decimal (6,2),

check (disponibilita >= 0 and prezzo >= 0)

);

create table cataloghi(

nome varchar(30) not null references articoli(nome) on update cascade on delete no action,

numero\_pag int default 0,

rilegatura varchar (40),

check (numero\_pag >= 0),

primary key(nome)

);

create table scontrini(

id integer primary key,

data datetime not null default current\_timestamp,

costo decimal (6,2) not null default 0,

visitatore integer not null references visitatori(id) on update cascade on delete set null,

check (costo >= 0)

);

create table contenuti(

articolo varchar(30) not null references articoli(nome) on update cascade on delete no action,

scontrino integer not null references scontrini(id) on update cascade on delete cascade,

quantita int not null default 1,

check (quantita >= 1),

primary key(articolo, scontrino)

);

create table autori(

id integer auto\_increment primary key,

nome varchar(20) not null,

cognome varchar(20) not null,

data\_nasc date,

luogo\_nasc varchar(20),

sesso character not null check (sesso = ‘M’ or sesso = ‘F’)

);

create table scritture(

autore integer not null references autori(id) on update cascade on delete cascade,

catalogo varchar(30) not null references cataloghi(nome) on update cascade on delete cascade,

primary key(autore, catalogo)

);

create table ordini(

id integer primary key,

data datetime not null default current\_timestamp,

costo decimal (6,2) not null default 0,

is\_spedito bit not null default 0,

check (costo >= 0)

);

create table carrelli(

ordine integer not null references ordini(id) on update cascade on delete cascade,

catalogo varchar(30) not null references cataloghi(nome) on update cascade on delete no action,

quantita int not null default 1,

check (quantita >= 1),

primary key(ordine, catalogo)

);

create table questionari(

visitatore integer not null references visitatori(id) on update cascade on delete cascade,

tipologia\_visita enum('visita guidata', 'visita libera', 'partecipazione ad una mostra/laboratorio'),

accompagnatori\_visita enum('solo', 'partner/coniuge', 'famiglia', 'amici/parenti/conoscenti', 'scolaresca', 'gruppo organizzato'),

motivazione\_visita enum('vedere oggetti belli', 'incontrare persone con interessi simili ai miei', 'vedere oggetti importanti', 'imparare cose nuove', 'trascorrere tempo libero con amici/parenti', 'approfondire le mie conoscenze', 'passare un momento personale piacevole'),

titolo\_studi enum('Elem. ', 'Lic. Media', 'Diploma', 'Laurea/PostLaurea', 'Nessuno'),

numero\_visite int check (numero\_visite >= 0),

ritorno enum('Assolutamente no', 'Probabilmente No', 'Probabilmente Si', 'Assolutamente Si'),

valutazione\_visita int check(valutazione\_visita >= 0 and valutazione\_visita <= 10),

valutazione\_esperienza int check(valutazione\_esperienza >= 0 and valutazione\_esperienza <= 10),

valutazione\_struttura int check(valutazione\_struttura >= 0 and valutazione\_struttura <= 10),

primary key(visitatore)

);

create table clienti(

id integer not null references visitatori(id) on update cascade on delete cascade,

nome varchar(20) not null,

cognome varchar(20) not null,

tessera bit not null default 0,

cellulare varchar(15) unique,

email varchar(25) unique,

primary key (id)

);

Create table eventi(

id integer auto\_increment primary key,

data\_inizio datetime not null

);

create table mostre(

evento integer not null references eventi(id) on update cascade on delete cascade,

num\_opere int not null default 0 check (num\_opere >= 0),

descrizione varchar(500),

titolo varchar(30) not null unique,

tema varchar(30),

artista varchar (30),

data\_fine date,

primary key(evento)

);

create table tipi\_clientela(

nome varchar(30) primary key

);

create table esposizioni(

mostra integer not null references mostre(evento) on update cascade on delete cascade,

opera integer not null references opere(id) on update cascade on delete cascade,

primary key (mostra, opera)

);

create table attivita(

evento integer not null references eventi(id) on update cascade on delete cascade,

is\_tour\_guidato bit not null,

capienza int not null default 30,

check (capienza >= 0),

primary key (evento)

);

create table prenotazioni(

gruppo integer not null references gruppo(id) on update cascade on delete cascade,

attivita integer not null references attivita(evento) on update cascade on delete cascade,

data\_prenotazione datetime not null default current\_timestamp,

primary key (gruppo, attivita)

);

create table illustrazioni(

attivita integer not null references attivita(evento) on update cascade on delete cascade,

guida integer not null references guide(id) on update cascade on delete cascade,

primary key (attivita, guida)

);

Create table laboratori\_reali(

attivita integer not null references attivita(evento) on update cascade on delete cascade,

laboratorio varchar(30) not null references laboratori(nome) on update cascade on delete no action,

piano int,

stanza varchar(3),

primary key(attivita)

);

create table laboratori(

nome varchar(30) primary key,

tipologia enum( 'Design ', 'MuseoAperto '),

durata time,

costo\_persona decimal (6,2) not null default 0 check (costo\_persona >= 0),

costo\_classe decimal (6,2) not null default 0 check (costo\_classe >= 0)

);

create table indirizzamenti(

laboratorio varchar(30) not null references laboratori(nome) on update cascade on delete cascade,

tipo\_clientela varchar(30) not null references tipi\_clientela(nome) on update cascade on delete cascade,

primary key (laboratorio, tipo\_clientela)

);

Create table copie(

id integer not null references opere(id) on update

cascade on delete cascade,

numero integer not null,

is\_calco bit default 1,

scala varchar(10),

primary key(id)

);

Create table modelli\_architettonici(

copia integer not null references copie(id) on update

cascade on delete cascade,

is\_smontabile bit default 0,

is\_pieno bit default 0,

primary key(copia)

);

Create table abbonamenti(

cliente integer not null references clienti(id),

data\_rilascio datetime not null default current\_timestamp,

data\_scadenza date,

costo decimal (6,2) not null default 0,

tipologia enum('mensile', 'trimestrale', 'semestrale', 'annuale') not null,

primary key (cliente, data\_rilascio)

);