UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Programmazione Web

Linee guida per il Progetto #1

DOCENTE
Paolo Fosci

EMAIL
paolo.fosci@unibg.it

TITOLARE DEL CORSO
Prof. Giuseppe Psaila

Gruppi di progetto

- Gruppi di 2 o 3 persone
- Form per iscrizione del gruppo: https://forms.office.com/e/DT1aHeJpBJ
- In via eccezionale sono consentiti casi singoli (scrivetemi una mail)
- Il form rimane aperto fino a fine Aprile 2025

- I primi progetti saranno comunicati entro la metà di Aprile 2025

Linee guida generali

- Per il primo progetto dell'anno ad ogni gruppo verrà dato lo schema di un database differente.
  - Dovete crearlo sulla vostra piattaforma e popolarlo in maniera massiva (usate pure fonti esterne, o create dati sintetici con qualunque sistema riteniate utile – **l'importante è avere una certa numerosità e varietà nei dati**)
- Dovrete sviluppare un'applicazione per la quale vi verrà fornito un template di interfaccia (uno per gruppo)
- Ogni applicazione dovrà realizzare delle pagine per cercare dati per ognuna delle tabelle del database
- Laddove siano i dati siano linkabili, creare i suddetti link.
- Ogni applicazione dovrà realizzare tutte le operazioni di CRUD per una tabella particolare che vi verrà indicata.

Data Base – Linee guida

1. Di seguito vi vengono dati i dettagli (contesto e specifica funzionale corredati di schema concettuale e logico) di ogni database di riferimento.
2. Creare ogni tabella che vedete nello **schema logico** e popolarla rispettando le relazioni tra le tabelle.
3. Quando create il database non importa se nel database impostate o meno le chiavi e/o i vincoli referenziali, questi li dovrete gestire in caso tramite l'applicazione
4. Le tabelle sulle quali creare una pagina di ricerca sono quelle azzurre negli schemi concettuali.

Data Base – Elenco

Elenco dei database di riferimento (nelle pagine seguenti la descrizione di ognuno di essi:

#	Database	Riferimento
1	Museo	Ex 1
2	Telefoni	Ex 2
3	Servizio Sanitario	Ex 3
4	Ricettario	Ex 4
5	Cinema	Ex 5
6	Social Network	Ex 6

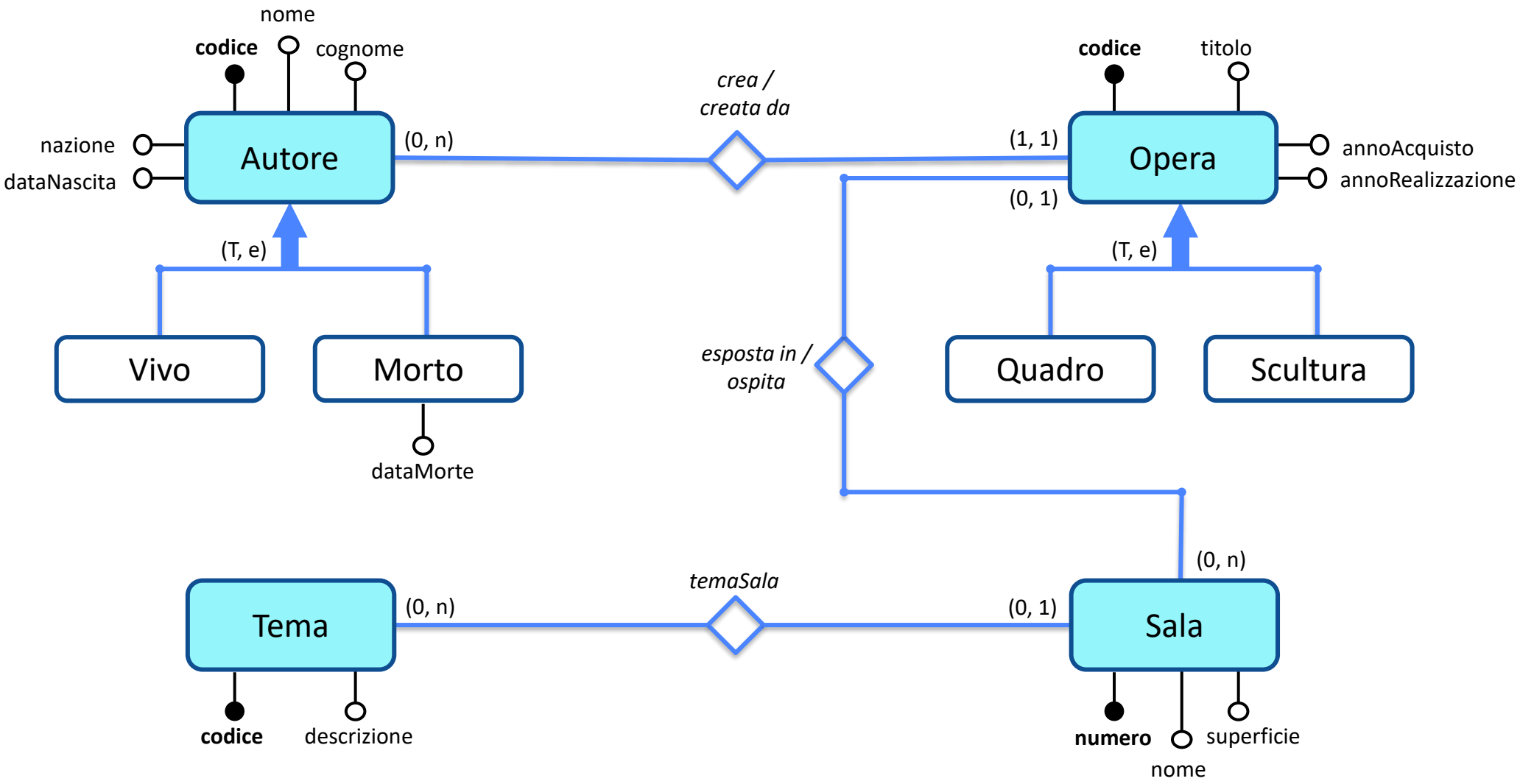
#	Database	Riferimento
7	Ombrelloni	Ex 7
8	Autostrade	TE 08-06-2018
9	Motorizzazione	TE 09-09-2019
10	Distribuzione Acqua	TE 20-01-2020
11	Quiz	TE 03-07-2020
12	Palestra	TE 29-06-2021

DB1 - Ex 1: Museo

Base di dati per la gestione delle opere in un museo.

Il museo possiede un insieme di opere che, a seconda dei periodi, possono essere o meno mostrate nelle sale.
La base dati deve descrivere la situazione corrente, cioè quali sono le opere mostrate e in quali sale.
Un'opera è identificata da un codice ed è caratterizzata dal titolo, dall'anno di realizzazione, dall'anno di acquisto da parte del museo e dall'autore; le opere si suddividono in quadri e sculture. Gli autori sono opportunamente descritti nella base dati: sono identificati da un codice e caratterizzati dal cognome, dal nome, dalla nazione e dalla data di nascita; per gli autori deceduti si vuole avere anche la data della morte.
Non tutti gli autori sono necessariamente associati ad un'opera.
Il museo è fisicamente organizzato in sale: ogni sala è identificata da un numero, e caratterizzata dal nome e dai metri quadri. Alla singola sala può essere associato un tema, descritto da un apposito elenco; si noti che lo stesso tema può essere associato a diverse sale, così come potrebbe non essere associato a nessuna sala (perché, in quel momento, non vi sono opere esposte che potrebbero far riferimento a quel tema).
Un tema è identificato da un codice ed è caratterizzato da una descrizione (del testo).
Infine, si vuole sapere quali opere sono esposte nelle varie sale; una stessa opera può essere esposta in una sala, ma potrebbe essere non esposta (quindi, essere implicitamente nel magazzino).

DB1 - Ex 1: Museo



DB1 - Ex 1: Museo

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Autore (**codice**, nome, cognome, nazione, dataNascita, tipo, dataMorte*)

Constraint su Autore:
(tipo='vivo' **AND** dataMorte **IS NULL**) **OR**
(tipo='morto' **AND** dataMorte **IS NOT NULL**)

Opera (**codice**, autore, titolo, annoAcquisto, annoRealizzazione, tipo, espostaInSala*)

Constraint su Opera:
tipo='Quadro' **OR** tipo='Scultura'

Sala (**numero**, nome, superficie, temaSala*)

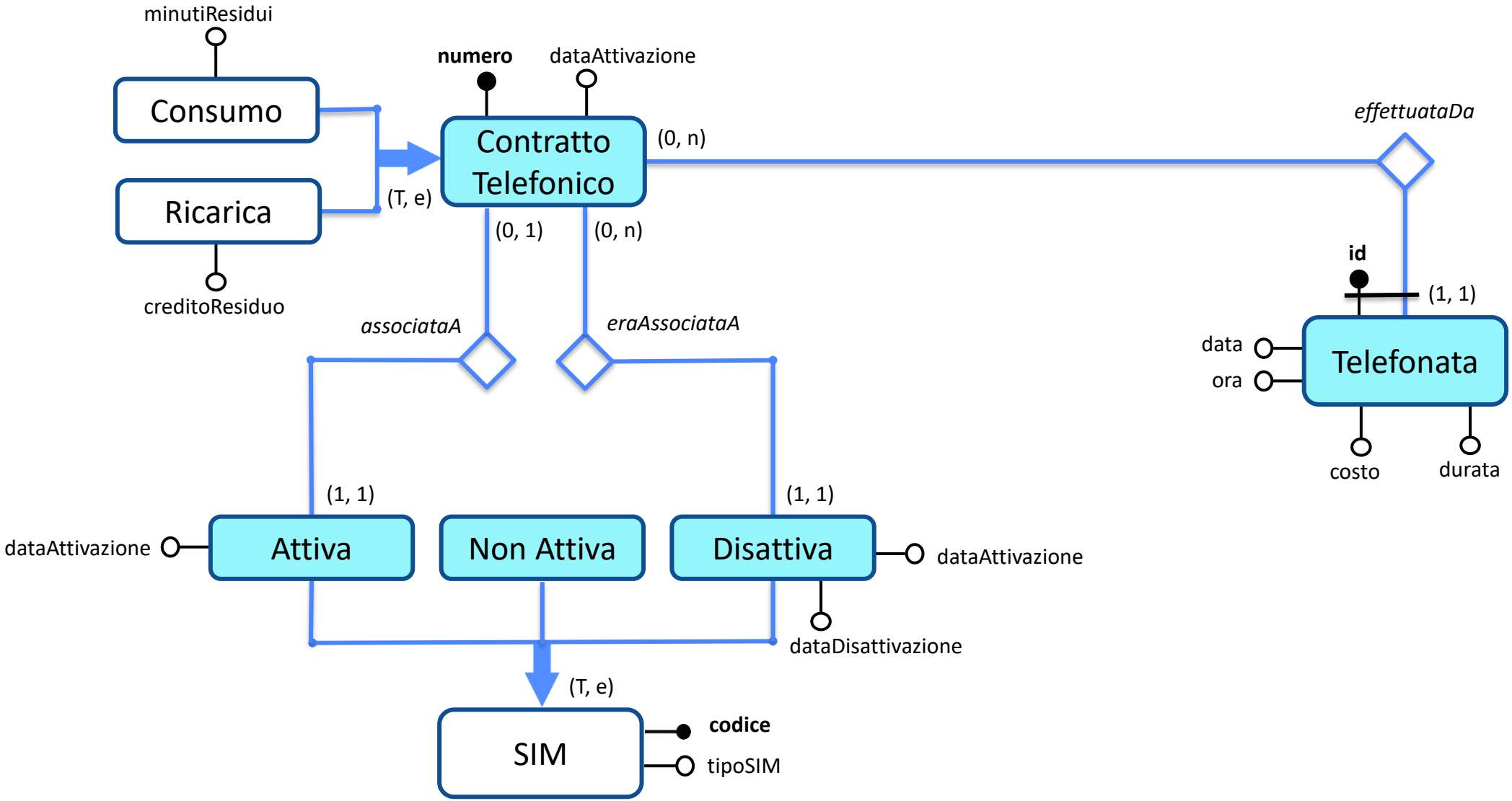
Tema (**codice**, descrizione)

DB2 - Ex 2: Contratti Telefonici

Base di dati per la gestione dei contratti telefonici di un operatore mobile.

Il cuore del servizio è il contratto telefonico. Questo viene identificato dal numero di telefono ed è caratterizzato dalla data di attivazione del numero. In particolare, i contratti telefonici sono suddivisi in contratti a ricarica e contratti a consumo: per i primi, si ha il credito residuo; per i secondi, si hanno i minuti residui del mese.
Ai contratti è normalmente associata una SIM la quale è identificata da un codice ed è caratterizzata dal tipo. Le SIM attualmente associate ad un contratto sono attive. Per le SIM attive si vuole sapere la data di attivazione. Si noti che possono esserci contratti temporaneamente senza SIM e che alcune SIM posso essere state disattivate. In questo caso, è di interesse conoscere oltre a quella che era la data di attivazione, anche la data di disattivazione e quale era il contratto telefonico su cui erano state attivate.
Infine, per ogni contratto si registrano alcune informazioni sulle telefonate effettuate. Una telefonata è identificata da un numero univoco nell'ambito del contratto, e caratterizzata dalla data e ora, dalla durata, dal costo (deliberatamente non si vuole sapere il numero di telefono chiamato).

DB2 - Ex 2: Contratti Telefonici



DB2 - Ex 2: Contratti Telefonici

Telefonata (**id**, **effettuataDa**, data, ora, durata, costo)

ContrattoTelefonico (**numero**, dataAttivazione, tipo, minutiResidui*, creditoResiduo*)

Constraint su ContrattoTelefonico:
(tipo='consumo' **AND** minutiResidui **IS NOT NULL**
AND creditoResiduo **IS NULL**)
OR
(tipo='ricarica' **AND** minutiResidui **IS NULL**
AND creditoResiduo **IS NOT NULL**)

SIMAttiva (**codice**, tipoSIM, associataA, dataAttivazione)
Indice senza duplicati su SIMAttiva (associataA)

SIMDisattiva (**codice**, tipoSIM, eraAssociataA,
dataAttivazione, dataDisattivazione)

SIMNonAttiva (**codice**, tipoSIM)

Legenda:
- ***sottolineato***: *attributo chiave*
- ***** : *attributo annullabile*

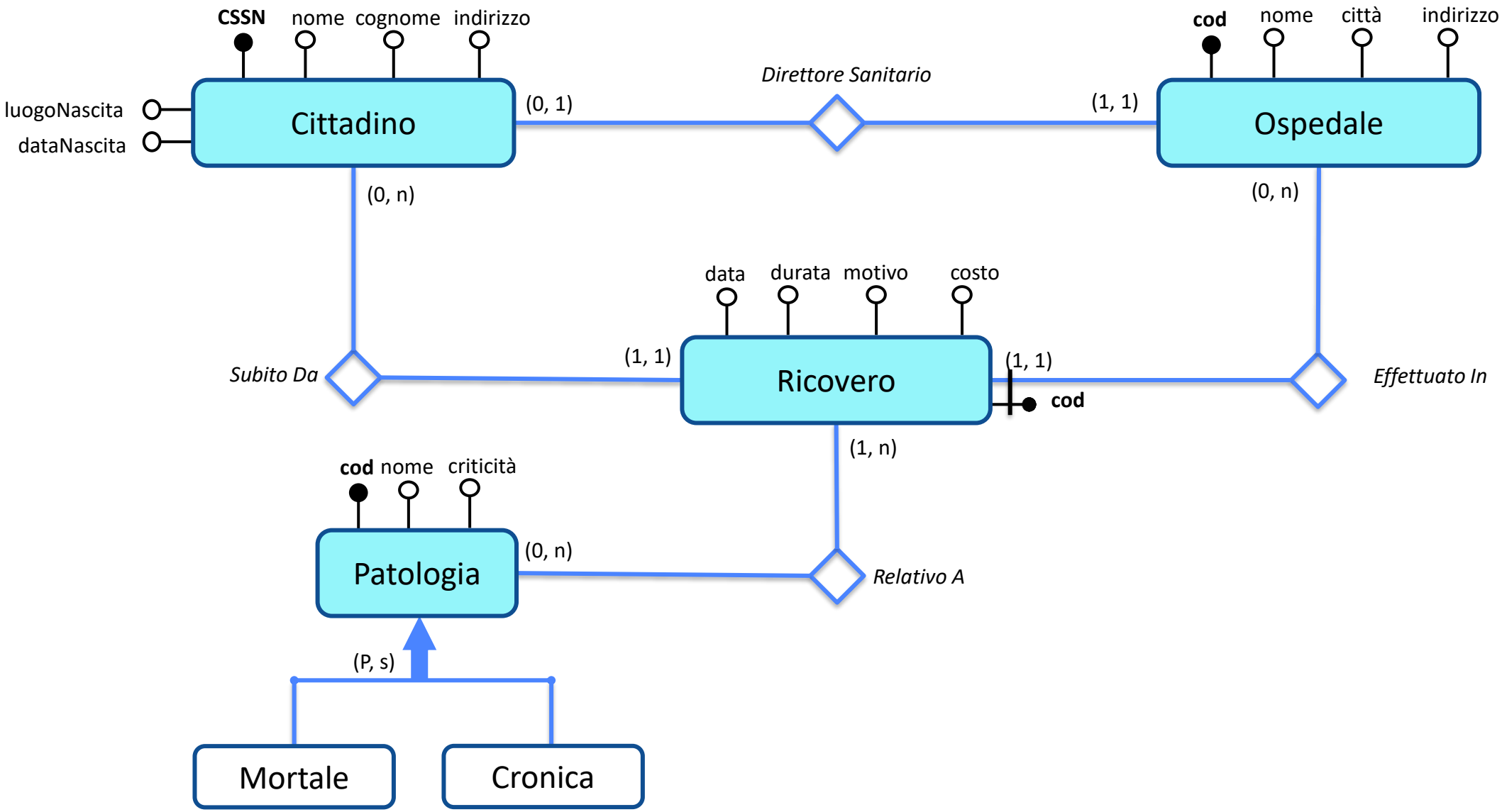
Nota per la creazione dei dati e per il CRUD:
Gli insiemi delle chiavi nelle tabelle *SIMAttiva*, *SIMDisattiva* e *SIMNonAttiva* devono essere disgiunti!

DB3 - Ex 3: Servizio Sanitario

Base di dati di una Regione per la gestione delle informazioni sui ricoveri ospedalieri.

I cittadini sono noti a priori alla Regione, in quanto hanno il codice del Servizio Sanitario Nazionale. Quindi, un cittadino è identificato dal codice suddetto ed è caratterizzato dagli usuali dati anagrafici.
Anche gli ospedali sono noti a priori alla Regione: un codice li identifica e sono poi caratterizzati dal nome, dalla città, dall'indirizzo e dal nome del Direttore Sanitario.
Una persona può essere al massimo Direttore Sanitario di un solo ospedale.
La Regione riceve le informazioni sui ricoveri: il ricovero è identificato da un codice univoco per l'ospedale nel quale viene effettuato, ed è caratterizzato dalla data di inizio, dai giorni, dal motivo e dal costo, nonché dal cittadino (paziente) ricoverato.
I ricoveri avvengono per curare una o più patologie, che sono note a propri: ogni patologia è identificata dal codice, ed è caratterizzata dal nome, e da un livello di criticità (tipicamente, un numero).
In particolare, si vogliono gestire due sottoinsiemi: quello delle patologie mortali e quello delle patologie croniche (i due sottoinsiemi potrebbero essere non disgiunti e sicuramente non sono esaustivi).

DB3 - Ex 3: Servizio Sanitario



DB3 - Ex 3: Servizio Sanitario

Legenda:
- ***sottolineato***: *attributo chiave*
- ***** : *attributo annullabile*

Cittadino (**CSSN**, nome, cognome, dataNascita, luogoNascita, indirizzo)

Ospedale (**codice**, nome, città, indirizzo, direttoreSanitario)

Indice senza duplicati su Ospedale (direttoreSanitario)

Ricovero (**codOspedale**, **cod** , paziente, data, durata, motivo, costo)

PatologiaRicovero (codOspedale, codRicovero , codPatologia)

Patologia (**cod**, nome, criticità)

PatologiaCronica (**codPatologia**)

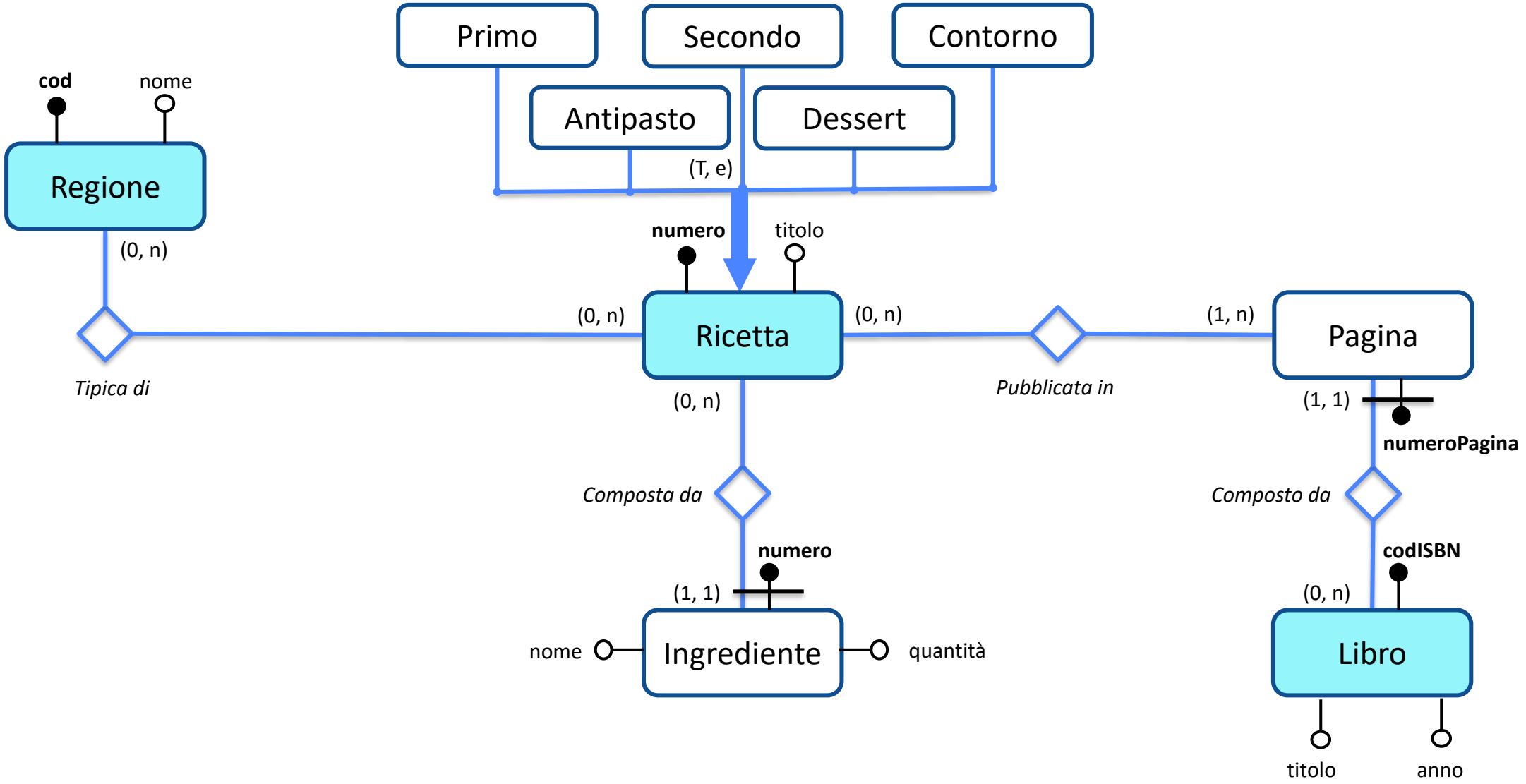
PatologiaMortale (**codPatologia**)

DB4 - Ex 4: Ricettario

Base di dati di una Regione per la gestione delle informazioni sui ricoveri ospedalieri.

L'editore che pubblica i libri deve avere una raccolta di ricette. Ogni ricetta è identificata da un numero ed ha un titolo; le ricette vengono suddivise in cinque categorie: antipasti, primi, secondi, contorni, dessert.
Le ricette possono essere tipiche di una o più regioni: in tal caso si vuol sapere quali sono queste regioni (dove le regioni sono identificate da un codice e caratterizzate da un nome).
Le ricette sono fatte da ingredienti: per ogni ricetta, vi è una serie di ingredienti, dove ogni ingrediente è identificato da un numero progressivo proprio della ricetta, ed è caratterizzato dal nome dell'ingrediente e dalla quantità.
Infine, le ricette vengono pubblicate nei libri.
Ogni libro è identificato dal suo codice ISBN ed è caratterizzato dal titolo e dall'anno di pubblicazione.
Ogni libro è composto da pagine, dove ogni pagina ha un numero proprio del particolare libro;
in una pagina, vengono pubblicate una o più ricette.

DB4 - Ex 4: Ricettario



DB4 - Ex 4: Ricettario

Legenda:
- ***sottolineato***: *attributo chiave*
- ***** : *attributo annullabile*

Regione (**cod**, nome)

RicettaRegionale (regione, ricetta)

Ricetta (**numero**, titolo, tipo)

Constraints su Ricetta:
tipo='antipasto' **OR** tipo='primo' **OR**
tipo='secondo' **OR** tipo='contorno' **OR** tipo='dessert'

Ingrediente (**numeroRicetta**, **numero**, ingrediente, quantità)

RicettaPubblicata (numeroRicetta, libro, numeroPagina)

Pagina (**libro**, **numeroPagina**)

Libro (**codISBN**, titolo, anno)

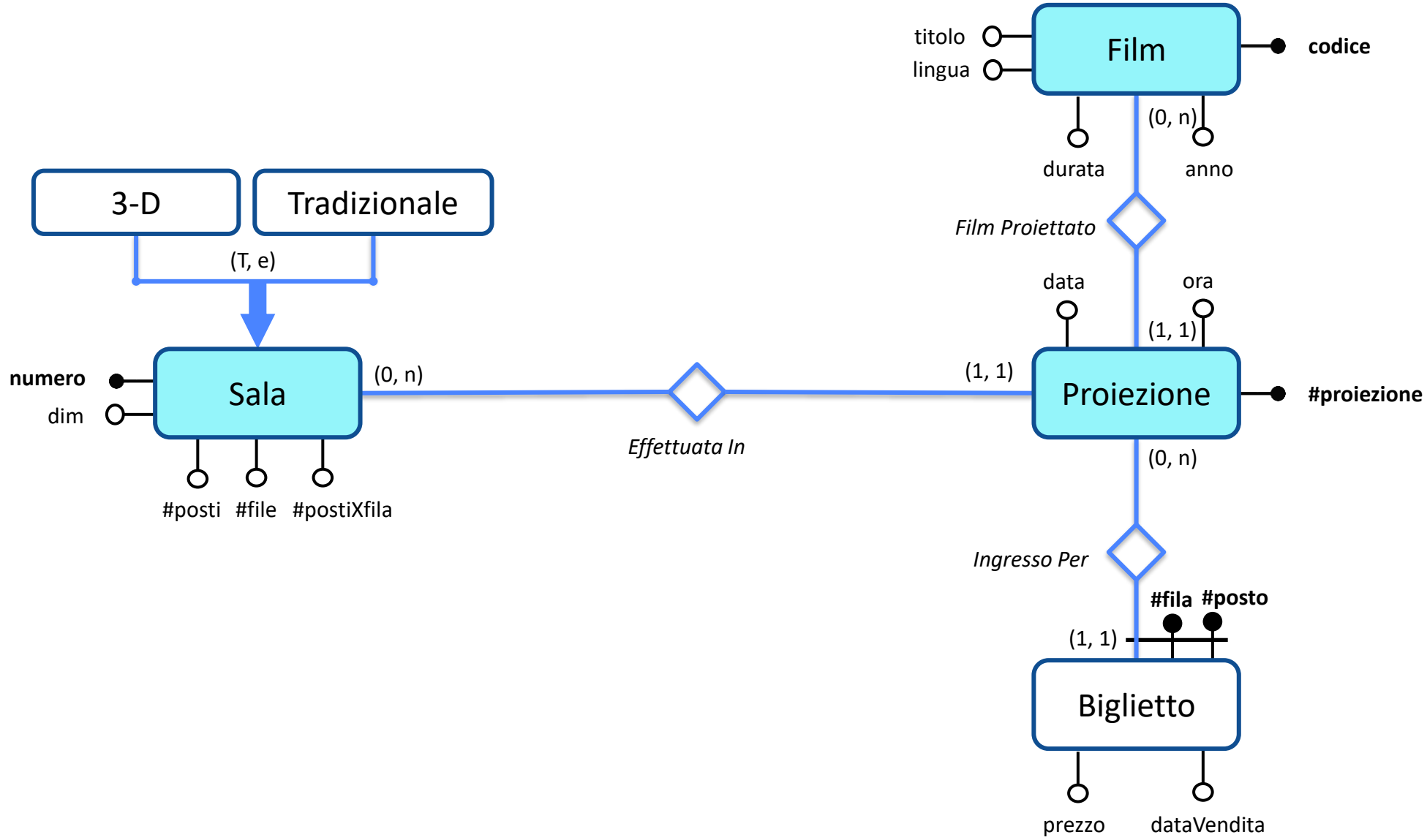
Nota per le Query:
Dei libri si vogliono sapere il numero delle pagine e anche delle ricette.
Delle ricette si vogliono sapere il numero dei libri dove sono pubblicate (e se ci riuscite fate vedere in quali libri)

DB5 - Ex 5: Cinema

Base di dati per la gestione dei biglietti di un cinema multisala.

Il cinema è dotato di diverse sale; ogni sala è identificata da un numero ed è caratterizzata dal numero di posti, dalla larghezza dello schermo (in pollici), dal numero di file e dal numero di posti per fila; le sale vengono suddivise poi in sale per 3D e sale tradizionali.
Per poter organizzare la vendita dei biglietti, è necessario considerare i dati delle proiezioni. Ogni proiezione è identificata da un numero progressivo, e caratterizzata dalla data e dall'ora della proiezione, dalla sala nella quale la proiezione viene fatta, nonché dal film proiettato, del quale si vogliono avere alcune informazioni, quali il titolo, l'anno, la lingua, la durata (si consideri che lo stesso film può essere oggetto di diverse proiezioni).
Per gestire la vendita dei biglietti, occorre, per ogni proiezione, fare in modo che lo stesso posto non venga occupato da più di una persona. Pertanto, un biglietto è identificato dal numero di proiezione, dal numero di fila e dal numero di posto nella fila. Inoltre, un biglietto è caratterizzato dalla data di vendita e dal prezzo pagato.

DB5 - Ex 5: Cinema



DB5 - Ex 5: Cinema

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Sala (**numero**, numPosti, dim, numFile, numPostiPerFila, tipo)

Constraints su Sala:
tipo='3-D' **OR** tipo='tradizionale'

Proiezione (**numProiezione**, sala, filmProiettato, data, ora)

Biglietto (**numProiezione**, **numFila**, **numPosto**, dataVendita, prezzo)

Film (**codice**, titolo, anno, durata, lingua)

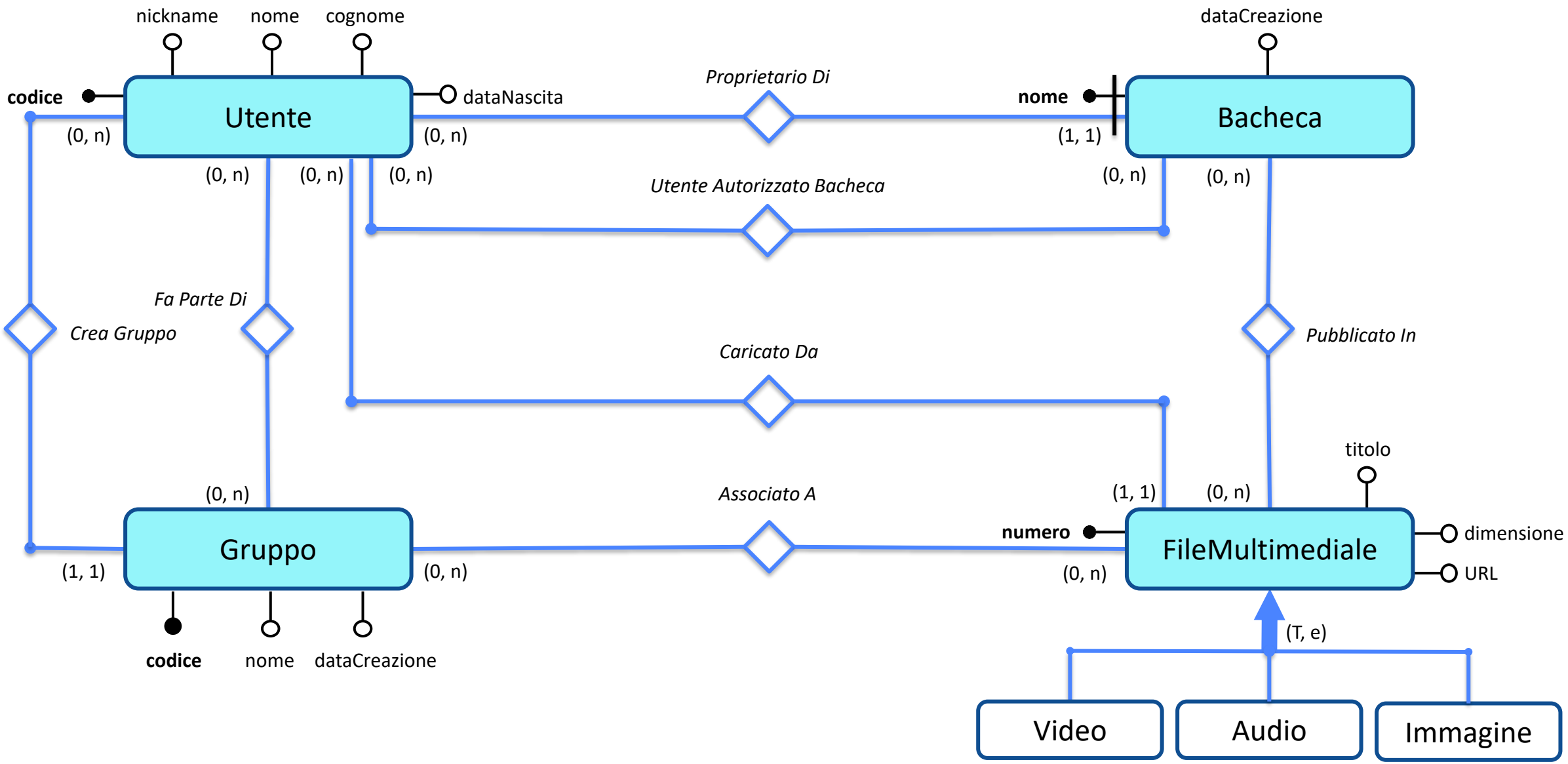
Nota per le Query:
Delle proiezioni si vuole anche il titolo del film, la sala e in numero dei biglietti venduti.

DB6 - Ex 6: Social Network

Base di dati per la gestione di un social network per scambio di file multimediali.

Il social network che stiamo considerando consente agli utenti di creare bacheche nelle quali possono inserire file multimediali caricati da loro stessi o da altri utenti. Gli utenti possono anche creare gruppi, nei quali possono essere pubblicati dei file multimediali. Vediamo nel dettaglio.
Gli utenti sono identificati da un codice e sono caratterizzati dal nickname e dagli usuali dati anagrafici. Un utente può avere associate delle bacheche, che crea egli stesso: una bacheca ha un nome che la identifica e una data di creazione (il nome è univoco per ciascun utente). Il proprietario di una bacheca può autorizzare altri utenti a vedere il contenuto della bacheca. Si noti che un utente potrebbe decidere di non creare alcuna bacheca.
All'interno di una bacheca possono essere pubblicati dei file multimediali. Questi sono identificati da un numero e sono caratterizzati da un titolo, dalla dimensione, dall'URL del file (relativo ai sistemi del social network) e dall'utente che ha caricato il file nei sistemi per la prima volta; inoltre vengono suddivisi in video, audio e immagini. Si osservi che lo stesso file multimediale può essere pubblicato in bacheche diverse. Per finire, gli utenti possono creare dei gruppi.
Un gruppo è identificato da un codice ed è caratterizzato dal nome, dalla data di creazione e dall'utente che lo ha creato. L'utente creatore autorizza altri utenti a far parte del gruppo. Al gruppo, possono essere associati dei file multimediali, allo scopo di condividerli tra tutti gli utenti del gruppo.

DB6 - Ex 6: Social Network



DB6 - Ex 6: Social Network

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Utente (**codice**, nickname, nome, cognome, dataNascita)

UtenteAutorizzatoBacheca (*utenteAutorizzato*, *codUtente*, *nomeBacheca*)

Bacheca (**codiceUtente**, **nome**, dataCreazione)

FilePubblicatoBacheca (*file*, *codUtente*, *nomeBacheca*)

FileMultimediale (caricatoDa, **numero**, titolo, dimensione, URL, tipo)

Constraint su FileMultimediale:
tipo='audio' **OR**
tipo='video' **OR**
tipo='immagine'

UtenteAutorizzatoGruppo (*codUtente*, *codGruppo*)

Gruppo (creatoDa, **codice**, nome, dataCreazione)

FileAssociatoGruppo *codGruppo*, *file*)

DB7 - Ex 7: Noleggio Ombrelloni

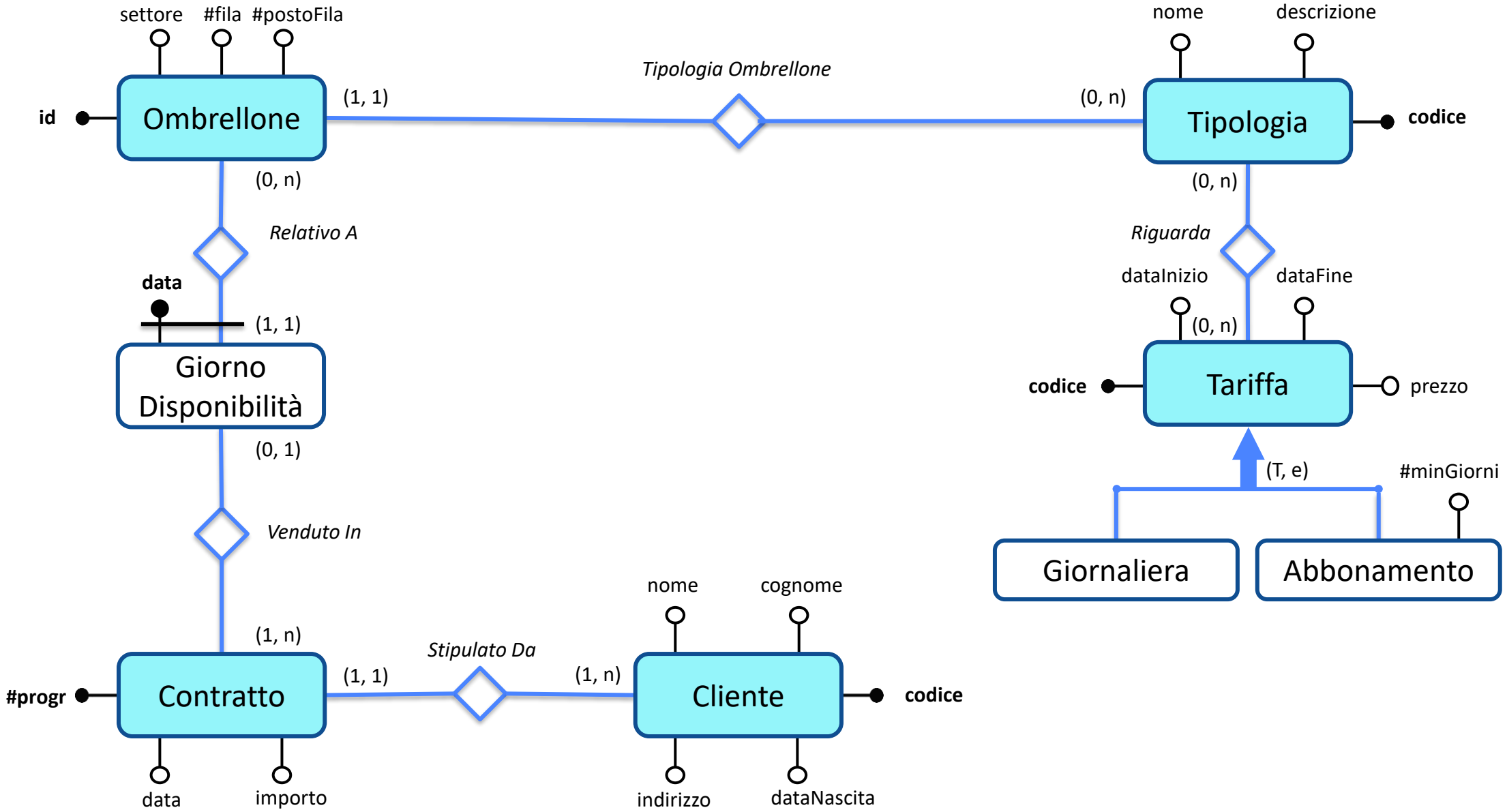
Base di dati per la gestione del noleggio degli ombrelloni in una spiaggia attrezzata in cui si ha la necessità di gestire l'affitto degli ombrelloni ai clienti, in base al tipo di ombrellone e al periodo.

Ogni ombrellone è identificato da un identificatore numerico, ed è caratterizzato dal settore della spiaggia, dal numero di fila e dal numero d'ordine all'interno della fila. Gli ombrelloni sono associati ad una tipologia, dove ogni tipologia è identificata da un codice ed è caratterizzata da un nome e dalla descrizione (testuale) degli accessori in dotazione agli ombrelloni di quella tipologia (per esempio, sdraio, lettino, ecc.). Per ogni tipologia, si ha un insieme di tariffe associate: le tariffe indicano quale prezzo applicare a seconda del periodo e del tipo di affitto che viene scelto dal cliente. Pertanto, una tariffa è identificata da un codice ed è caratterizzata dal periodo di validità della tariffa, nonché dal prezzo; inoltre, le tariffe vengono suddivise in giornaliere (che valgono per un affitto di un solo giorno) o in abbonamento e per queste ultime si vuole sapere il numero minimo di giorni per far decorrere l'abbonamento.

Per poter affittare gli ombrelloni senza correre il rischio di affittare lo stesso ombrellone a due clienti contemporaneamente, occorre predisporre, per ciascun ombrellone, un insieme di giorni di disponibilità: ogni giorno di disponibilità è identificato univocamente dalla data rispetto all'ombrellone di riferimento (ovviamente, possono esserci giorni di disponibilità con la stessa data ma per ombrelloni diversi).

Per finire, l'ufficio vendite effettua un contratto di affitto con un cliente; il contratto è identificato da un numero progressivo ed è caratterizzato dalla data, dall'importo complessivo e dai giorni di disponibilità degli ombrelloni affittati con quel contratto (ad un giorno di disponibilità può essere associato al più un contratto).

DB7 - Ex 7: Noleggio Ombrelloni



DB7 - Ex 7: Noleggio Ombrelloni

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Ombrellone (**id**, settore, numFila, numPostoFila, tipologia)

Tipologia (**codice**, nome, descrizione)

TipologiaTariffa (***codTipologia***, ***codTariffa***)

Tariffa (**codice**, prezzo, dataInizio, dataFine, tipo, numMinGiorni*)

Constraint su Tariffa:
(tipo='Giornaliera' **AND** numMinGiorni **IS NULL**)
OR
(tipo='Abbonamento' **AND** numMinGiorni **IS NOT NULL**)

GiornoDisponibilita (**idOmbrellone**, **data**)

OmbrelloneVenduto (***idOmbrellone***, ***data***, *contratto*)

Contratto (**numProgr**, data, importo, stipulatoDa)

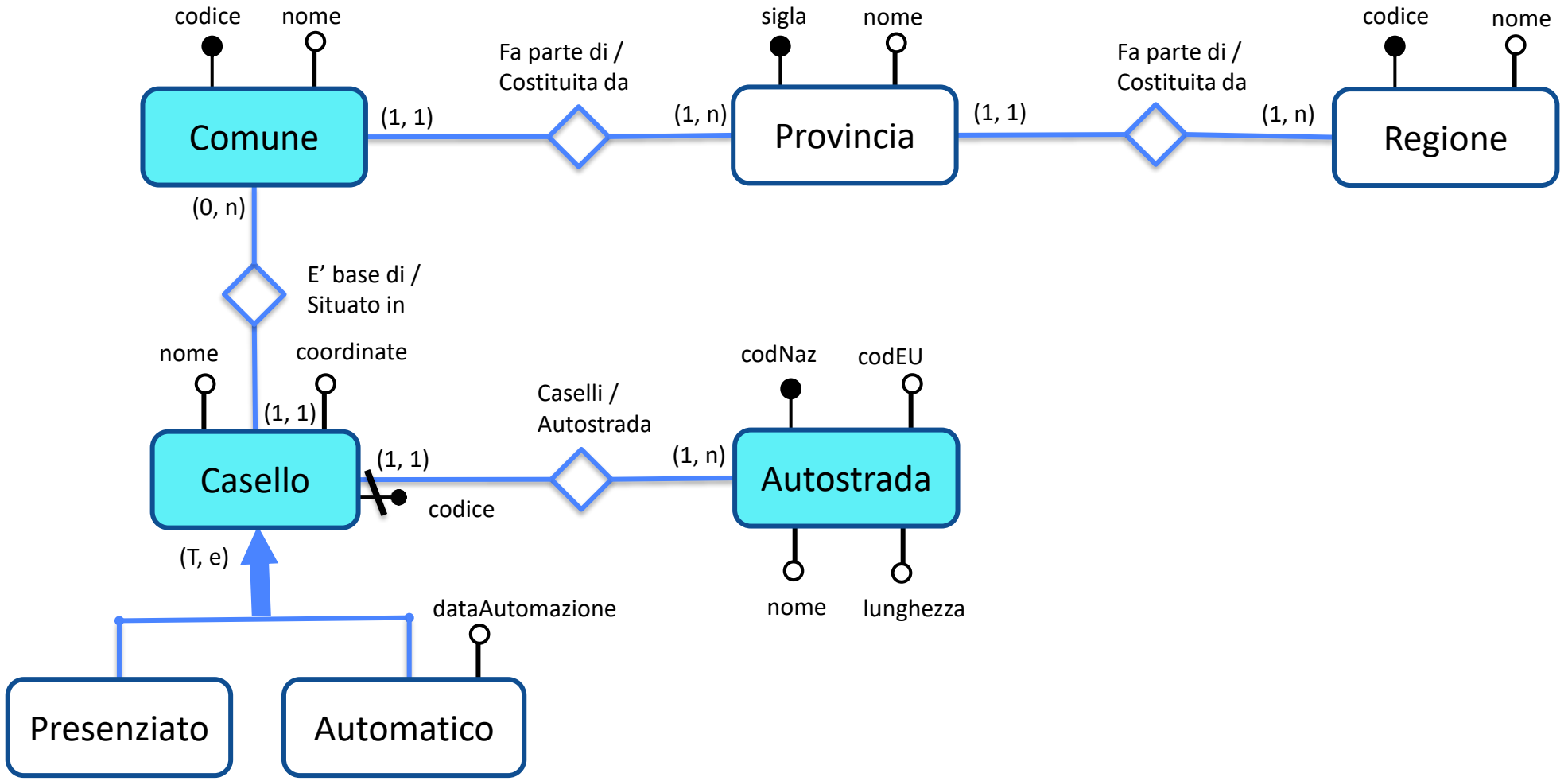
Cliente (**codice**, nome, cognome, dataNascita, indirizzo)

DB8 - Autostrade

Base di dati di un sistema autostradale.

La società ACME gestisce alcune autostrade sul territorio nazionale. Nella base dati ha bisogno di riportare le informazioni relative ai caselli e alla loro localizzazione sul territorio nazionale.
Per prima cosa, si considerino le informazioni relative ai comuni, alle provincie e alle regioni:
una regione è identificata da un codice ed è caratterizzata dal nome;
una provincia è identificata dalla sua sigla ed è caratterizzata dal nome e dalla regione in cui si trova;
un comune è identificato da un codice ed è caratterizzato dal nome e dalla provincia in cui si trova.
Veniamo quindi alle autostrade. Un'autostrada è identificata dal codice univoco nazionale ed è caratterizzato dal codice europeo, dal nome dell'autostrada e dalla lunghezza totale in chilometri.
I caselli sono identificati da un codice univoco per l'autostrada di cui fanno parte e sono caratterizzati dal nome del casello, dalle coordinate GPS e dal comune nel quale sono posizionati.
Infine, vengono suddivisi in automatici e presenziati (cioè con presenza degli operatori); per i caselli automatici, si vuole sapere in quale data sono stati resi totalmente automatici.

DB8 - Autostrade



DB8 - Autostrade

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Regione (**codice**, nome)

Provincia (**sigla**, regione, nome)

Comune (**codice**, provincia, nome)

Casello (**codice**, **autostrada**, comune, nome, coordinate, tipo, dataAutomazione*)

Autostrada (**codNaz**, codEU, nome, lunghezza)

Constraint su *Casello*:
(*tipo*='Automatico' AND *dataAutomazione* IS NOT NULL)
OR
(*tipo* ='Presenziato' AND *dataAutomazione* IS NULL)

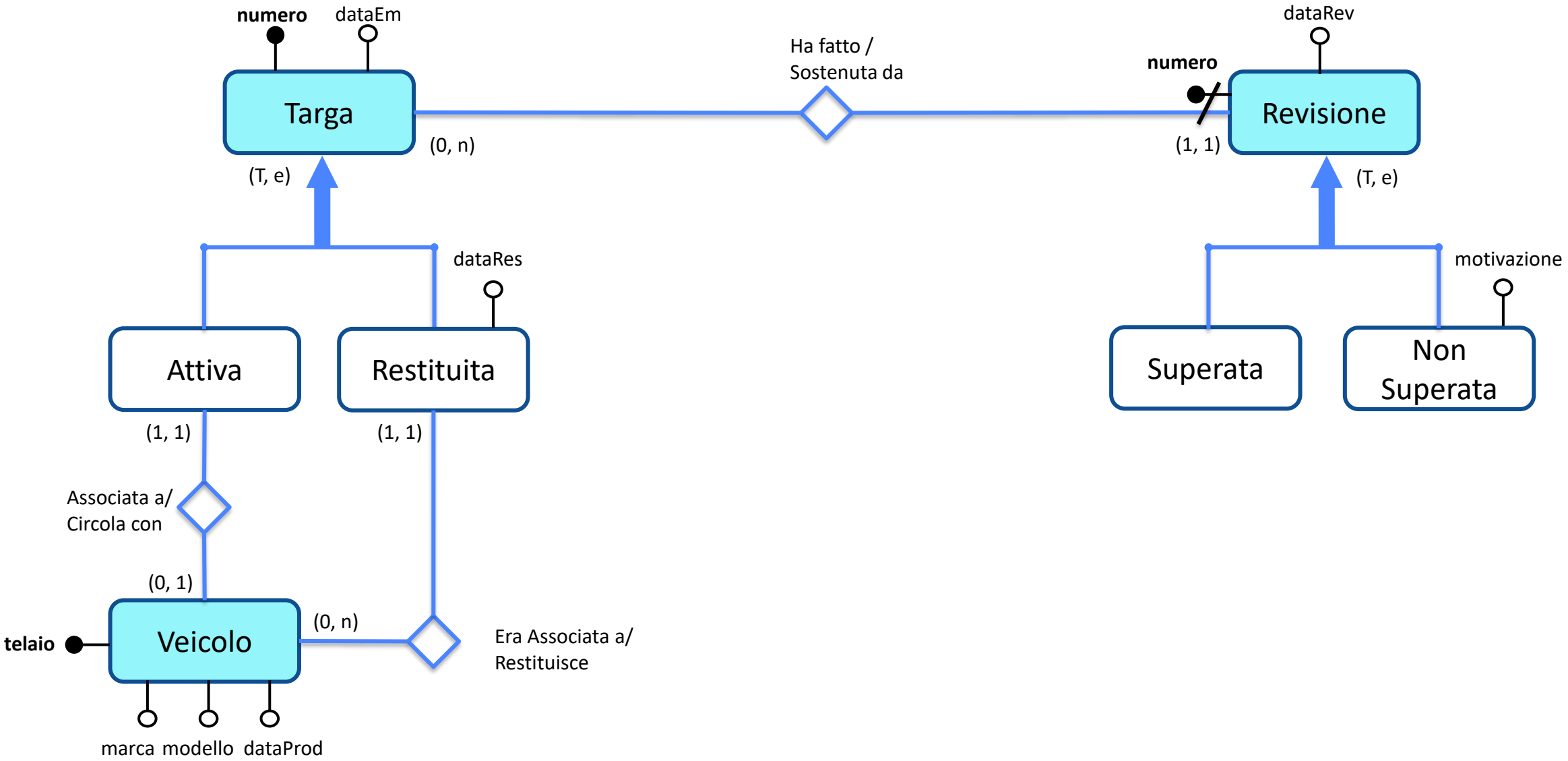
Note per le Query:
- Dei comuni indicare anche provincia e regione e # di caselli
- Delle autostrade indicare anche il numero dei caselli

DB9 - Motorizzazione

Si progetti la porzione di base di dati del sistema della Motorizzazione Civile per gestire i veicoli e le targhe.

Un veicolo esiste indipendente dal fatto di essere targato oppure no. Un veicolo è identificato dal numero di telaio ed è caratterizzato dalla marca, dal modello e dalla data di produzione.
Per poter circolare, un veicolo devo avere una targa, ma ci sono momenti in cui un veicolo può non avere la targa e nella sua storia può averne più d'una.
Quindi, una targa è identificata dal numero ed è caratterizzata dalla data di emissione. Le targhe sono suddivise in attive e restituite: per le targhe attive, si vuole sapere il veicolo associato (un veicolo, dal canto suo, può avere al più una targa attiva associata); per le targhe restituite, si vuole sapere la data di restituzione e il veicolo che era associato (dal canto suo, un veicolo può essere associato ad un numero arbitrario di targhe restituite).
I veicoli sono soggetti a revisione periodica, ma queste vengono riferite alla targa. Quindi, una revisione è identificata da un numero progressivo univoco rispetto alla targa; inoltre, una revisione è caratterizzata dalla data in cui avviene. Le revisioni vengono suddivise in superate e non superate: in quest'ultimo caso si vuole sapere il motivo per cui la revisione non è stata superata.

DB9 - Motorizzazione



DB9 - Motorizzazione

Revisione (**numero**, **targa**, dataRev, esito, motivazione*)

Targa (**numero**, dataEm)

TargaAttiva (**targa**, veicolo)

TargaRestituita (**targa**, veicolo, dataRes)

Veicolo (**telaio**, marca, modello, dataProd)

Constraint su ***Revisione***:
(***esito***='positivo' **AND** ***motivazione*** is NULL)
OR
(***esito***='negativo' **AND** ***motivazione*** is NOT NULL)

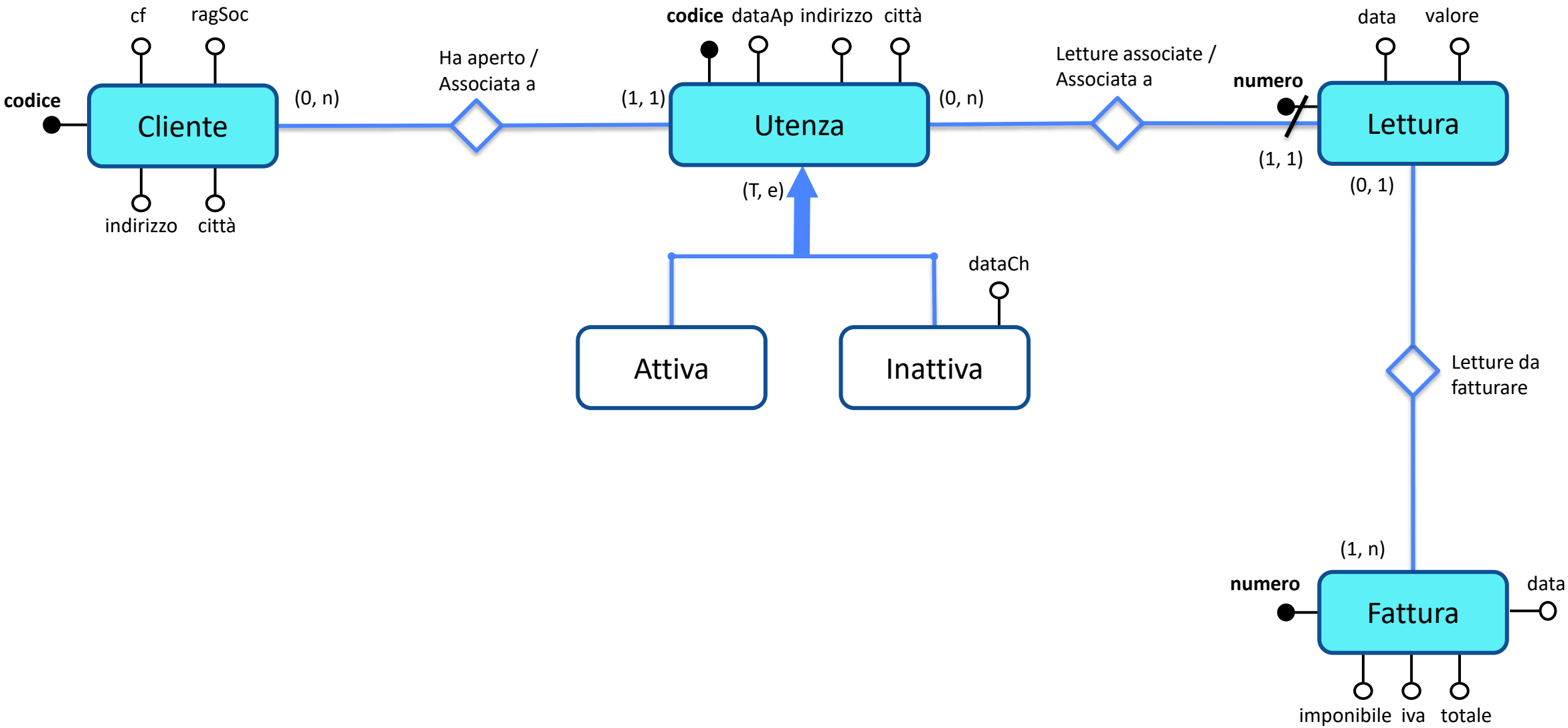
Indice su ***TargaAttiva***:
senza duplicati su ***veicolo***

DB10 – Distribuzione Acqua

Base di dati per la gestione del servizio di distribuzione dell'acqua.

L'ente di erogazione dell'acqua ha un insieme di utenze, ciascuna delle quali corrisponde ad un contatore. L'utenza è caratterizzata da un codice, dalla data di apertura, dall'indirizzo e dalla città; in particolare, un'utenza viene specializzata in attiva e inattiva, indicando per quest'ultima tipologia la data di disattivazione. Ogni utenza è associata ad un cliente: il cliente è caratterizzato da un codice, dalla ragione sociale, dall'indirizzo e dalla città, nonché dal codice fiscale.
Periodicamente, vengono effettuate delle letture presso le utenze; ogni lettura è identificata da un numero propria della particolare utenza alla quale è riferita ed è caratterizzata dalla data della lettura e dal valore letto sul contatore.
Alla fine, l'ente emette le fatture. Ogni fattura ha un numero univoco che la identifica, ed è caratterizzata dalla data, dall'imponibile, dall'IVA e dal totale da pagare, oltre che dalle letture che vengono fatturate con la fattura (si emette una fattura se vi sono delle letture da fatturare).

DB10 – Distribuzione Acqua



DB10 – Distribuzione Acqua

Legenda:
- ***sottolineato*** *: attributo chiave*
- ***** *: attributo annullabile*

Cliente (**codice**, cf, ragSoc, indirizzo, città)

Utenza (**codice**, cliente, dataAp, indirizzo, città, stato, dataCh*)

Lettura (**numero**, **utenza**, fattura*, data, valore)

Fattura (**numero**, data, imponibile, iva, totale)

Constraint su ***Utenza***:
(***stato***='attivo' AND ***dataCh*** is NULL)
OR
(***stato***='inattivo' AND ***dataCh*** is NOT NULL)

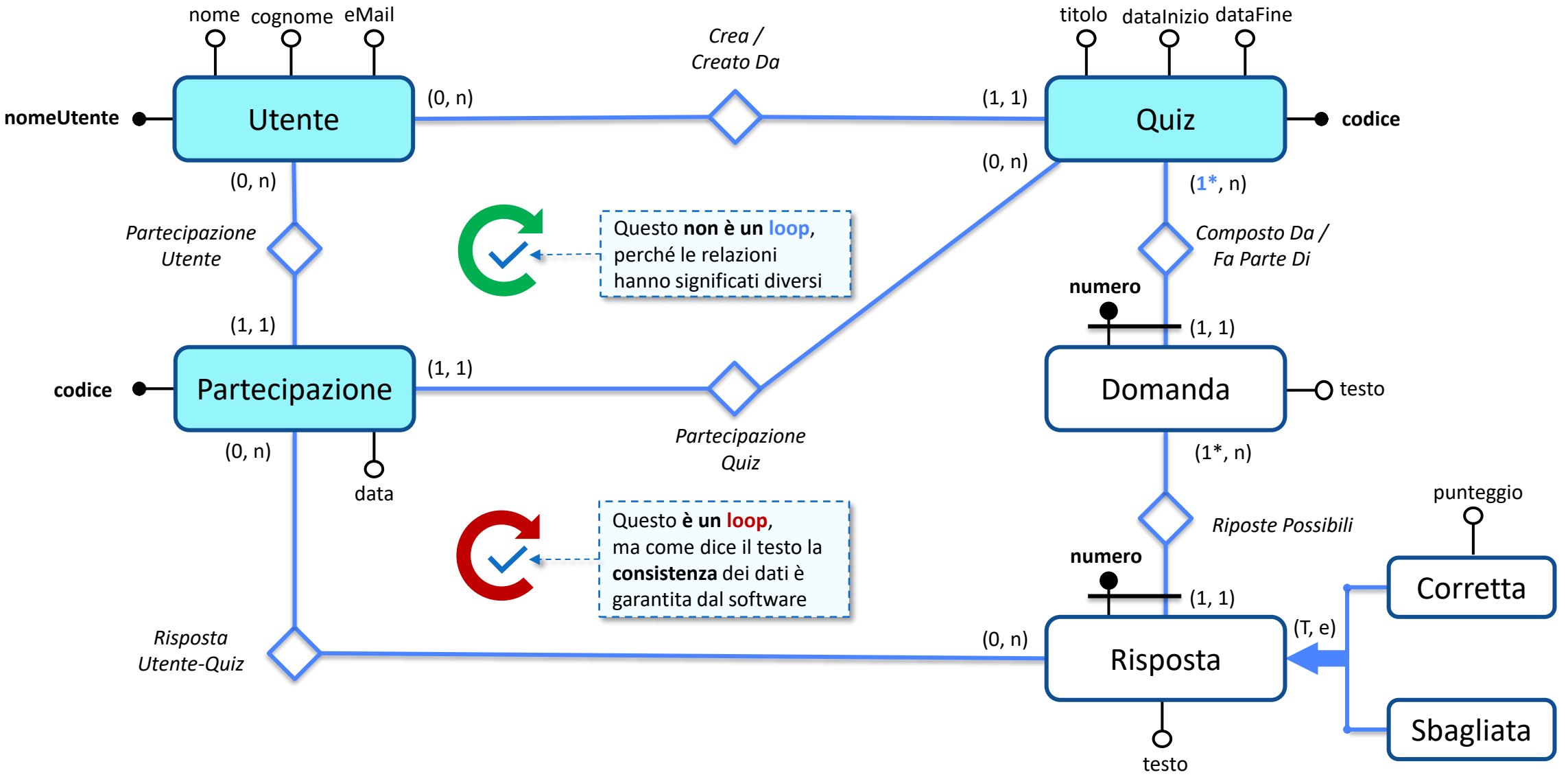
Nota per le Query:
- Visualizzare solo le letture per per Utenza o Fattura
- Di un cliente indicare quante utenze ha.
- Di un'utenza indicare quante letture ha.
- Di una fattura indicare quante letture ha.

DB11 – Quiz

Base di dati per la gestione di quiz online.

Il sistema web costruito al di sopra della base dati consente ai propri utenti di creare quiz (chiunque può creare dei quiz) e partecipare a quiz creati da altri utenti (chiunque può partecipare ad un quiz). Gli utenti sono identificati da un nome-utente e sono caratterizzati dal nome, dal cognome e dall'indirizzo email.
Un quiz è identificato da un codice ed è caratterizzato dal titolo, dalla data di apertura e dalla data di chiusura del quiz, oltre che dall'utente che lo ha creato. Un quiz è formato da domande. Ogni domanda è identificata da un progressivo proprio del quiz ed è caratterizzata dal testo della domanda. Poiché prendiamo in considerazione solo domande a risposta chiusa, le singole risposte sono, a loro volta, identificate da un numero di risposta, proprio della domanda; sono quindi caratterizzate dal testo della risposta. Le risposte si suddividono in sbagliate e corrette; queste ultime hanno un determinato punteggio (prevediamo il caso generale che una domanda possa avere più risposte corrette, ciascuna con un punteggio specifico).
Infine, gli utenti partecipano ad un quiz. Una partecipazione è identificata da un codice ed è caratterizzata dall'utente, dal quiz al quale partecipa, dalla data di partecipazione al quiz. Quindi, la partecipazione è associata alle risposte date: si osservi che la partecipazione viene attivata prima che l'utente inizi a scegliere le risposte e che sarà cura del software controllare che l'utente risponda solo a domande del quiz associato.

DB11 – Quiz


nome
cognome
eMail
Utente
nomeUtente
(0, n)
Crea /
Creato Da
(1, 1)
titolo
dataInizio
dataFine
Quiz
codice
(0, n)
(1*, n)
Partecipazione
Utente
(1, 1)
Questo non è un loop,
perché le relazioni
hanno significati diversi
Composto Da /
Fa Parte Di
numero
(1, 1)
codice
Partecipazione
(1, 1)
Domanda
testo
(0, n)
data
Partecipazione
Quiz
(1*, n)
Questo è un loop,
ma come dice il testo la
consistenza dei dati è
garantita dal software
punteggio
Riposte Possibili
numero
(1, 1)
Corretta
Risposta
Utente-Quiz
(0, n)
(T, e)
Risposta
Sbagliata
testo

DB11 – Quiz

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Utente (**nomeUtente**, nome, cognome, eMail)

Quiz (**codice**, creatore, titolo, dataInizio, dataFine)

Domanda (**quiz**, **numero**, testo)

Risposta (**quiz**, **domanda**, **numero**, testo, tipo, punteggio*)

Partecipazione (**codice** , utente, quiz, data)

RispostaUtenteQuiz (*partecipazione*, *quiz*, *domanda*, *risposta*)

Constraint su Risposta:
(tipo='Corretta' **AND** punteggio **IS NOT NULL**)
OR
(tipo='Sbagliata' **AND** punteggio **IS NULL**)

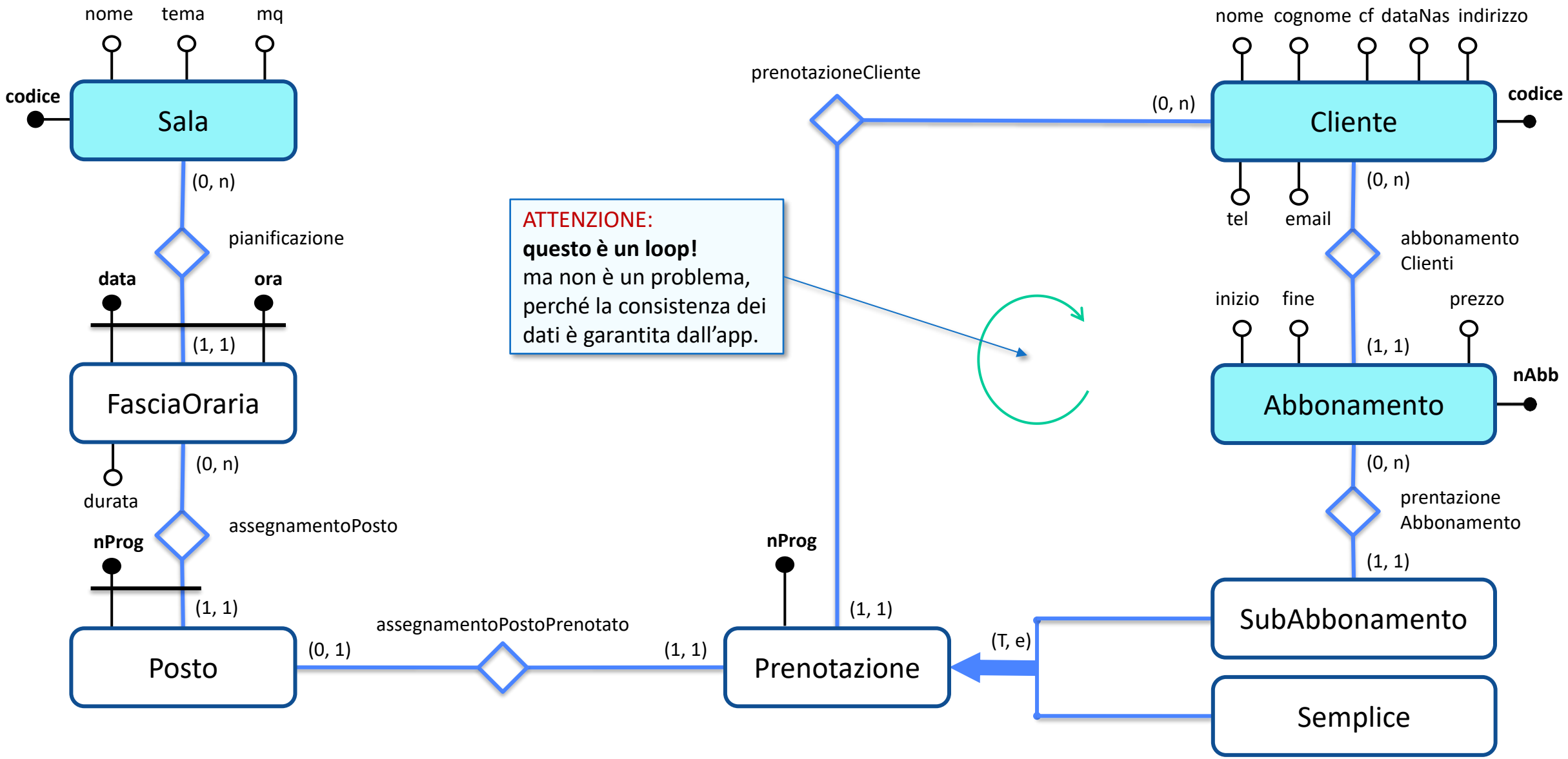
Note per le Query:
- Se dovete fare il CRUD del Quiz, la creazione è una funzionalità che parte dall'utente!
- Creare una pagina per il singolo quiz e mostrare le domande e le risposte possibili.
- **Extra**: creare una pagina per la singola partecipazione dove si fanno vedere le domande con le risposte disposte a caso

DB12 – Palestra

Base di dati per gestire l'accesso ad una palestra con le regole anti-COVID.

Il numero di clienti di una palestra che possono contemporaneamente entrare è limitato; inoltre, possono accedere solo su appuntamento.
La palestra è dotata di varie sale. Ogni sala è identificata da un codice ed è caratterizzata dai metri quadri, dal nome, dal tema della sala.
Per ogni sala, viene organizzato un calendario di fasce orarie, nelle quali i clienti possono prenotare e presentarsi. Una fascia oraria è identificata dalla data e dall'ora, in modo univoco per la sala; è inoltre caratterizzata dalla durata della fascia oraria. Per ogni fascia oraria, si definiscono i singoli posti prenotabili: ogni posto è identificato da un numero progressivo proprio della fascia oraria.
I clienti sono identificati da un codice e sono caratterizzati dagli usuali dati anagrafici. Se i clienti hanno sottoscritto un abbonamento, questo è identificato da un numero ed è caratterizzato dalla data di inizio e dalla data di fine, oltre che dal prezzo e dal cliente che lo ha sottoscritto.
Quando dall'app i clienti effettuano una prenotazione, questa viene registrata: essa è identificata da un numero progressivo ed è caratterizzata dal cliente e dal posto prenotato; ovviamente, un posto può essere associato al più ad una prenotazione. Le prenotazioni sono poi suddivise in "Semplici" o "Sub Abbonamento": in quest'ultimo caso, si vuole sapere quale abbonamento è stato usato.

DB12 – Palestra



DB12 – Palestra

Legenda:
- ***sottolineato*** : *attributo chiave*
- ***** : *attributo annullabile*

Sala (**codice**, nome, tema, mq)

FasciaOraria (**sala**, **data**, **ora**, durata)

Posto (**sala**, **data**, **ora**, **nProg**)

Prenotazione (**nProg**, cliente, sala, data, ora, posto)

Indice senza duplicati in ***Prenotazione*** su ***(sala, data, ora, posto)***

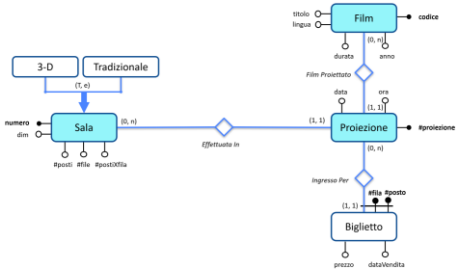
Cliente (**codice**, nome, cognome, cf, dataNas, indirizzo, tel, email)

Abbonamento (**nAbb**, cliente, inizio, fine, prezzo)

SubAbbonamento (**prenotazione**, abbonamento)

Es. di popolamento database

QUESTO E' UN ESEMPIO DI STRATEGIA PER POPOLARE IL DATABASE RELATIVO AL CASO DEL CINEMA:
PER I CASI SPECIFICI DEI VOSTRI PROGETTI FATE IPOTESI VEROSIMILI
PER AVERE DATI CONSISTENTI E VARIEGATI



- Cerchiamo una lista di film (magari qualche centinaio)
- Possiamo immaginare che il cinema abbia un certo numero di sale (es.: da 4 a 15?)
  - ognuna con un certo numero di posti (es: da 20 a 400) – gli altri dati inventateli
- Per le proiezioni immaginiamo un periodo di gestione del cinema (es: 3, 4 anni, tipo dal 2021 al 2025)
- Ipotizziamo che circa l'80% dei film venga proiettato. Poi per ogni film proiettato consideriamo
  - prima proiezione in un giorno a caso del nostro periodo di riferimento (spalmiamo i film nel periodo)
  - successivamente il film viene proiettato secondo una gaussiana da 1 a 21 giorni o qualcosa di simile
- Per ogni giorno di proiezione possiamo immaginare di aver un numero variabile di proiezioni in diversi orari (da 1 a 4 o 5) – in una certa sala scelta a caso
- **Volendo fare le cose bene,**
  controllare di non avere due proiezioni differenti nella stessa sala con stesso giorno e orario
- Per ogni proiezione, secondo una gaussiana, possiamo considerare un numero di biglietti in base alla capacità della sala

Data Base – Query

Le query di ricerca su una tabella devono prevedere diversi criteri (dipende dalla tabella). In generale:

- se una tabella A è legata da una relazione (0:1) o (1:1) ad un'altra tabella B sarebbe il caso di mostrare insieme alla riga di A i dati caratteristici di B (oltre all'ID). Es.: nel **DB1**, insieme ai dati di un' Opera mostrare il nome dell' Autore
- se una tabella A è legata da una reazione (0:n) o (1:n) ad un'altra tabella B allora mostrate il numero di entità di B legate ad ogni riga di A. Es.: nel **DB1**, insieme ai dati di un Autore mostrare il numero di Opere

Fate le vostre considerazioni caso per caso

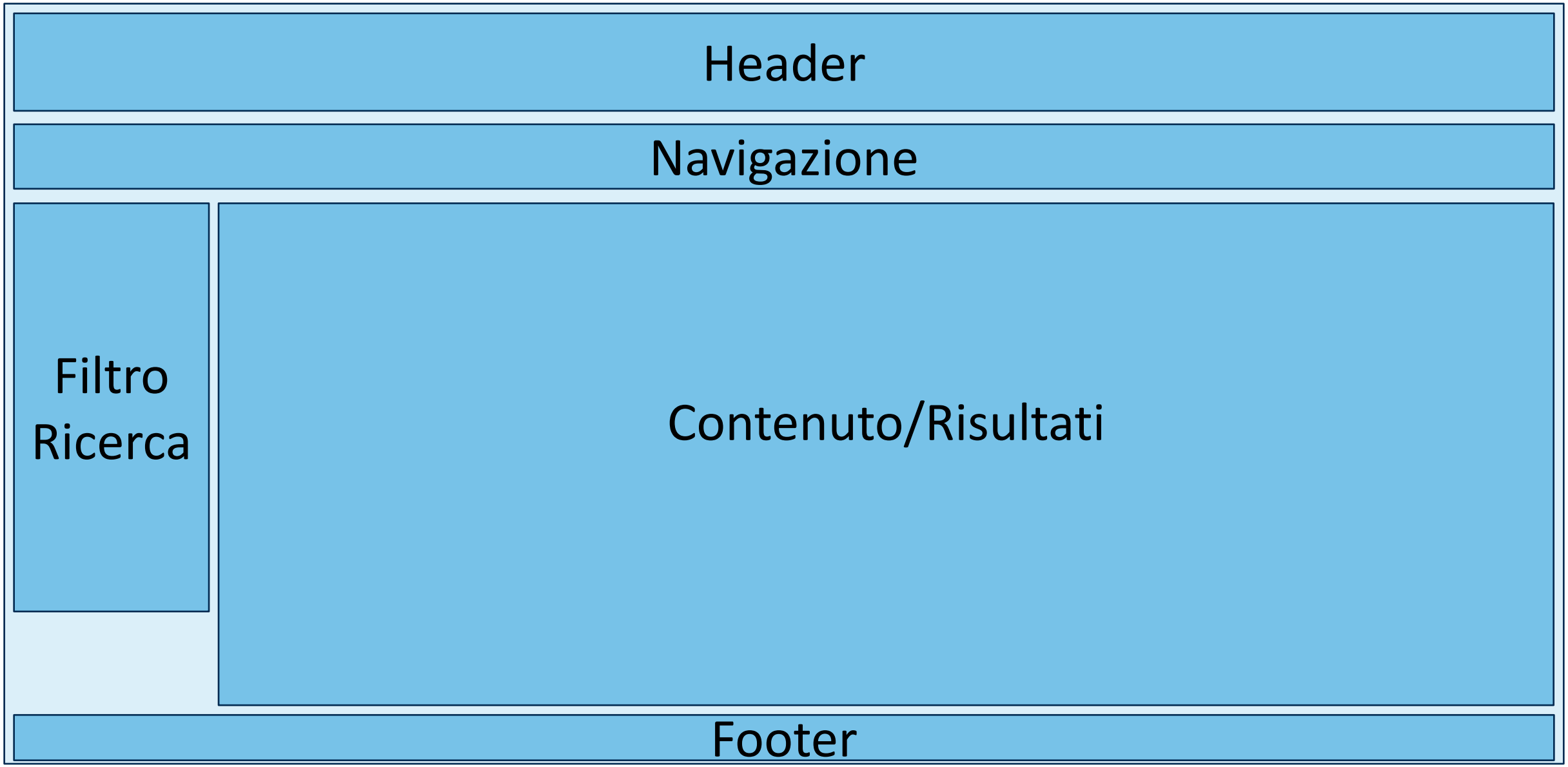
Interfacce

Di seguito sono elencati i template di riferimento per le interfacce.
I template sono solo di riferimento solo nel senso del posizionamento:
dimensioni, proprietà e colori sono a scelta vostra in base alle esigenze della vostra applicazione.
Se il template si adatta poco alla vostra applicazione (es: il form di ricerca richiede più spazio) fate le vostre valutazioni per eventuali modifiche.
Le interfacce devono essere il più robuste possibile a cambiamenti di dimensione del browser

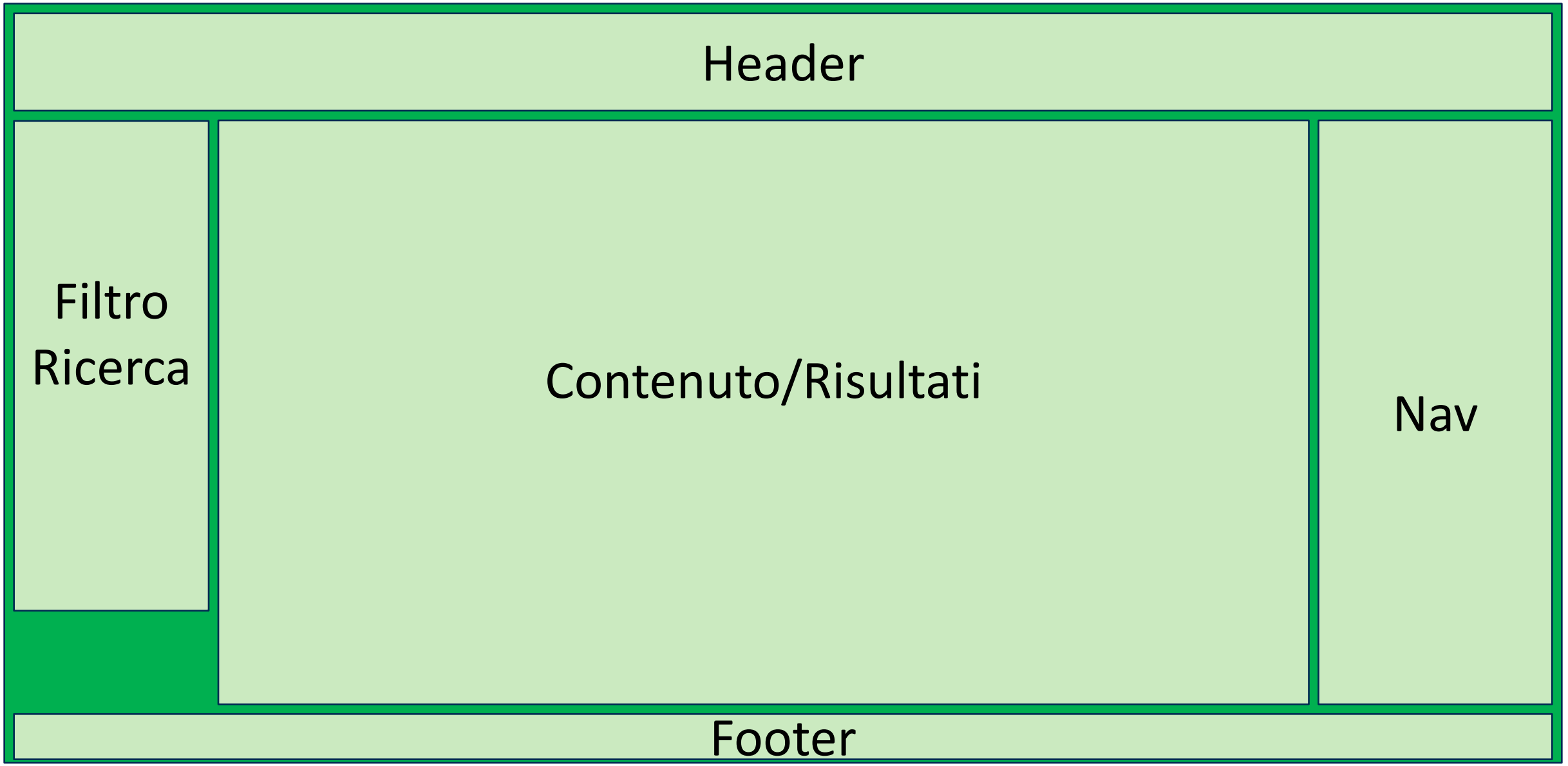
Vi verrà assegnata una palette di riferimento,
il che NON vuol dire che gli sfondi devono avere tutti una stessa tonalità!
Significa che dovete avere uno stile (per sfondi, caratteri o altri elementi tipo le icone) che richiami il colore della palette. Non vuol dire che tutti gli elementi devono avere certe tonalità, potete anche usare dei contrasti, anche forti, purché tutto sia coerente, e si capisca comunque qual è la tonalità dominante.

Potete usare anche delle immagini per gli sfondi e tutti gli elementi creativi ed effetti che ritenete di dover usare!

Template per Interfaccia 1



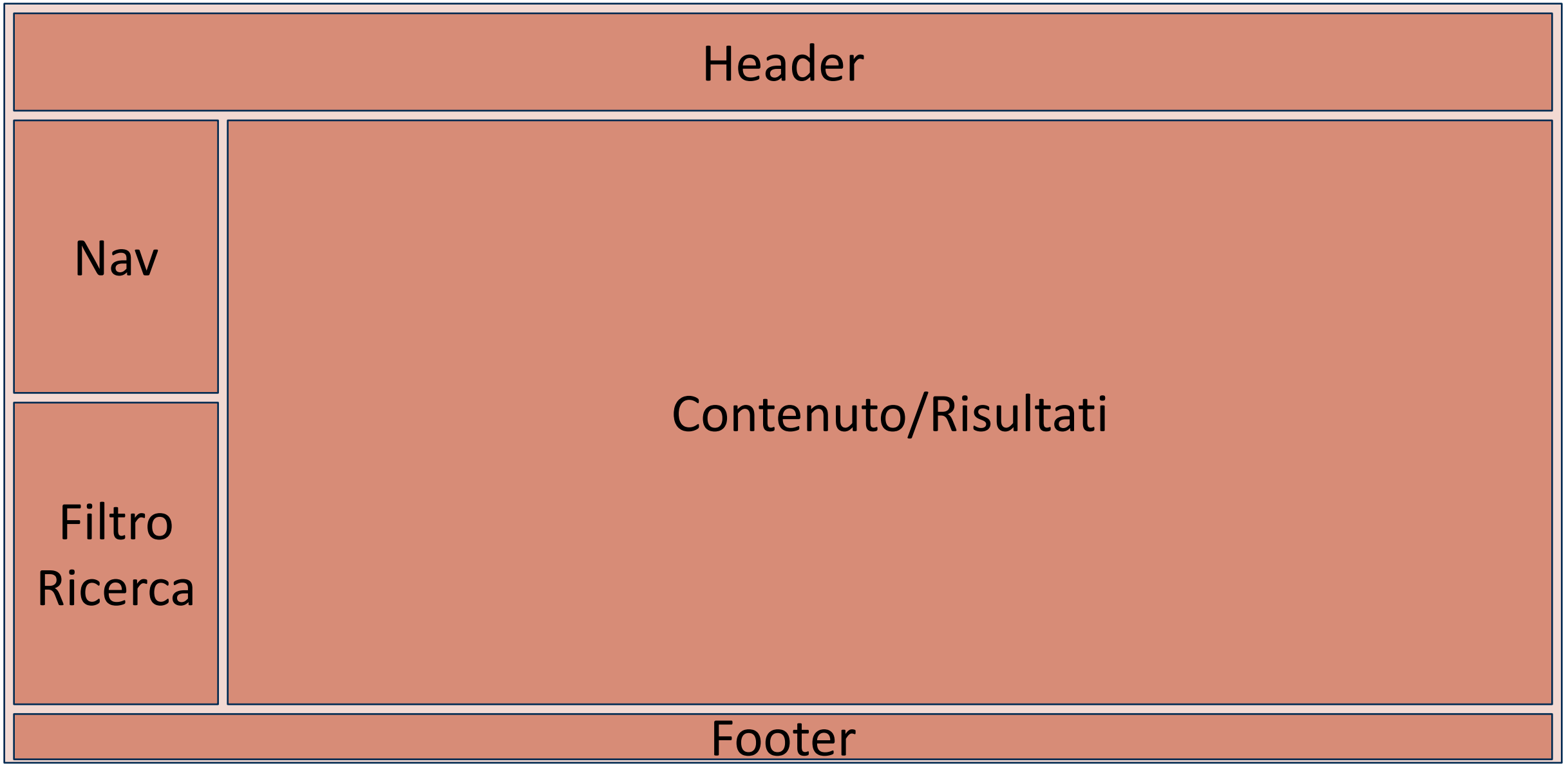
Template per Interfaccia 2



Template per Interfaccia 3

Template per Interfaccia 4

Template per Interfaccia 5



Link utili

- Flaticons
  https://www.flaticon.com/

- PNG Arts
  https://www.pngarts.com/it/

- Pixabay
  https://pixabay.com/

- Meat Ball Studios
  https://metaballstudios.creator-spring.com/

CRUD

Acronimo per tute quelle operazioni sui dati relative a:

- Create
- Read
- Update
- Delete

Cosa si aspetta l'utente medio? – E quello dummy?
Considerate che ci sono casi in cui alcune operazioni sono consentite solo a gruppi ristretti di utenti…

Create

Se l'utente deve inserire dei nuovi dati in database:

- che strumenti dobbiamo mettergli a disposizione?
- che messaggistica dobbiamo fornirgli?
- quali dati deve poter inserire?
  - gli ID deve fornirli oppure no?
  - come gestire eventuali chiavi esterne?

Read

Dobbiamo considerare due aspetti:

1. Che dati dobbiamo fornire all'utente?
   - L'utente deve essere conscio della struttura interna dei dati?
   - I dati mostrati all'utente debbono mantenere lo stesso formato interno?
2. Quali strumenti mettiamo a disposizione all'utente
   - Per cercare e filtrare i dati?
     - Le ricerche debbono essere stringenti oppure lasche?
   - Per ordinare i dati?

Update

Se l'utente deve modificare dei dati in un database:

- che strumenti dobbiamo mettergli a disposizione?
  - Ci sono funzionalità che possiamo sfruttare?
- che messaggistica dobbiamo fornirgli?
- quali dati deve poter modificare?
  - Ci sono dati che debbono essere protetti più di altri?

Delete

Se l'utente deve cancellare dei dati in un database:

- che strumenti dobbiamo mettergli a disposizione?
- che messaggistica dobbiamo fornirgli?
- quali dati deve poter cancellare?
- Che fare se ci sono dipendenze?
  - Gestiamo la cancellazione a cascata?

Messaggistica

Who, When, What, Where, How:

- A chi devo comunicare?
- Quando devo avvertire l'utente?
- Cosa debbo comunicargli?
- Dove?
- Come lo comunico?

Modalità di consegna

- Potete usare solo le tecnologie presentate finora: (HTML, CSS, JS, PHP, JQUERY, AJAX)
- Mail con [PW25] nell'oggetto insieme al nome del gruppo e del progetto
  - Tutti i componenti in cc
  - Link al sito
  - Zip file con il codice
  - Excel o Ods usato/i per popolare il database
  - Una paginetta di documentazione/spiegazione…

Per concludere…

Usate un po’ di fantasia, buon senso
e buon lavoro!

… e se avete problemi particolari scrivetemi una mail
(con il prefisso PW25)