

**SCUOLA POLITECNICA E DELLE SCIENZE DI BASE**

**DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL’INFORMAZIONE**

**CORSO DI LAUREA IN INFROMATICA**

**INSEGNAMENTO DI BASI DI DATI E SISTEMI OPERATIVI I**

**ANNO ACCADEMICO 2019/2020**

**Documentazione**

**Progettazione di una base di dati relazionale**

**<Sistema Gestionale per Recensioni Turistiche>**

**A cura di:**

Federico Gargiulo - N86002884

Antonio Garofalo - N86003129

**1 – Introduzione**

Il progetto che abbiamo scelto tra le tracce disponibili è stato quello di un ***Sistema Gestionale per Recensioni Turistiche,***si chiedeva al gruppo di due studenti di creare una base di dati che venisse gestita da un applicativo java.

I vincoli imposti, sono stati prettamente 2:

-Un utente può unicamente fare una singola recensione per una determinata location.

-Le location devo essere gestite in 3 macro-specializzazioni che sarebbero:

* Alloggio
* Attrazione
* Ristorante

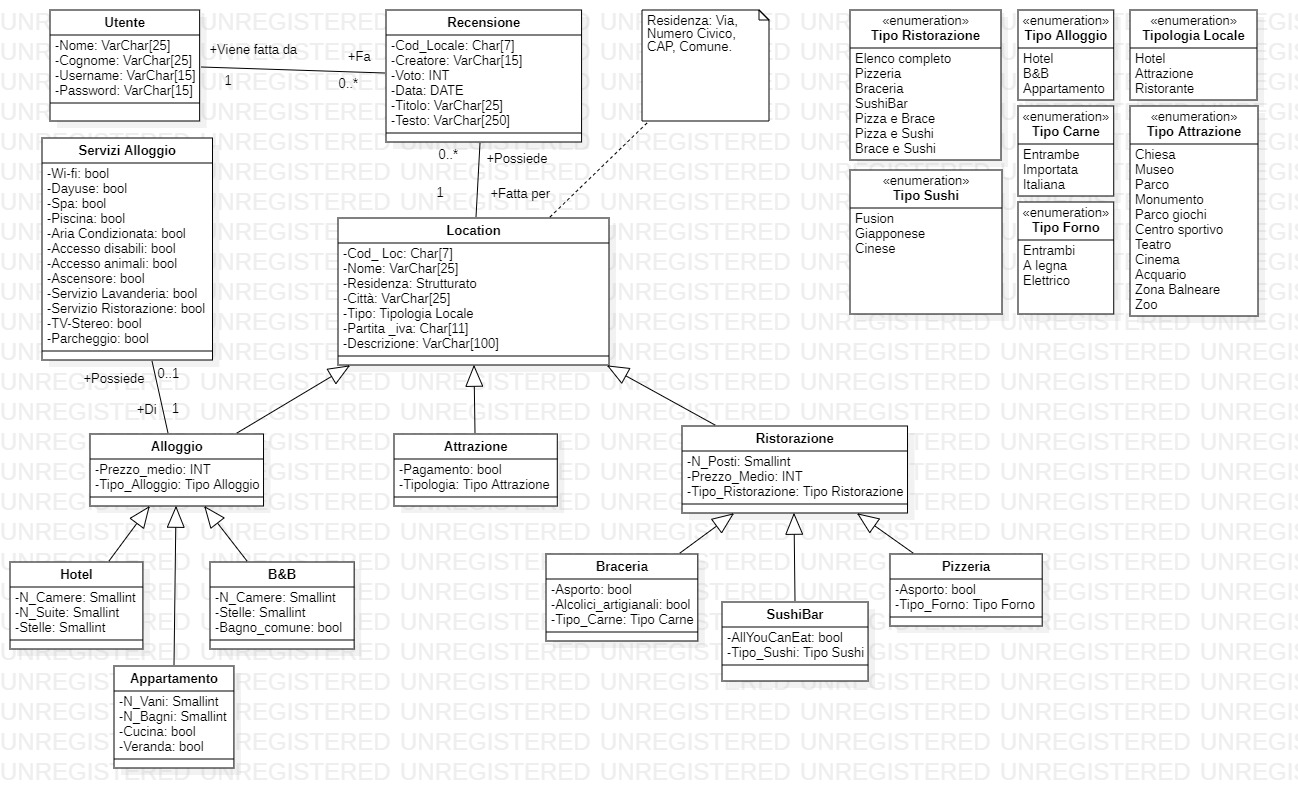
(N.B le specializzazioni devono essere  ulteriormente  raffinate)

E’ stato pensato di fare una base di dati su Oracle (precisamente versione 11g facilmente configurabile) gestendo il tutto con l’applicativo “SQL developer”.

La base di dati è stata modellata mediante Class Diagram con successiva revisione e ristrutturazione. Si noterà bene nella definizione di vincoli e tabelle la creazione anche di trigger che, per mancata presenza in Oracle, abbiamo inserito al fine di avere una base di dati dinamica che si aggiornasse automaticamente al cambiamento di specifici parametri e alcuni che velocizzassero gli inserimenti creando parallelamente altri inserimenti a catena da modificare con le informazioni desiderate.

Infine per i gruppi a 3 veniva chiesto un sistema di moderazione, tramite notifiche e controlli sulle recensione fatte prima di essere pubblicate.

**2 – Class Diagram (non ristrutturato)**



Il class diagram proposto per la soluzione del problema è formato da 13 classi (che diventeranno 14 con la ristrutturazione), 9 di queste classi sono specializzazioni di altre classi, per la precisione le classi sottostanti alla classe “Location” ad esclusione della classe “Servizi alloggio” sono specializzazioni.

Per raffinare meglio le classi si è optato per l’introduzione di vari nuovi tipi che sono sviluppati nelle enumerazioni in alto a destra.

Inoltre per completezza si è aggiunto un attributo strutturato (“Residenza” in “Location”) che verrà analizzato successivamente con la ristrutturazione del class diagram.

**3 – Modifiche di ristrutturazione**

Con un quadro generale avuto dal class diagram (non ristrutturato) si passa alle prime modifiche fatte in vista della ristrutturazione.

Ovviamente bisognava:

* Eliminare generalizzazioni/specializzazioni
* Eliminare attributi strutturati
* Eliminare eventuali molteplicità (assenti in principio)

Analizzando il precedente class diagram possiamo notare:

* Un attributo strutturato (in “Location)
* 9 specializzazioni e sotto-specializzazioni

La soluzione per il primo problema di ristrutturazione è stata scelta tra le soluzioni studiate durante il corso, ovvero, si è optato per la creazione di un “Entità” o classe totalmente estranea che crea un'unica associazione 1 a 0…\* con la classe “Location”, essa conterrà le residenze che esistono associate poi a nessuna o molte località.

Per il secondo problema, nonché quello principale richiesto anche dalla traccia si è optato per eliminare le generalizzazioni/specializzazioni lasciando intatta la struttura ad albero creatasi. Per questo motivo c’è stata la sostituzione completa con associazioni vincolate.

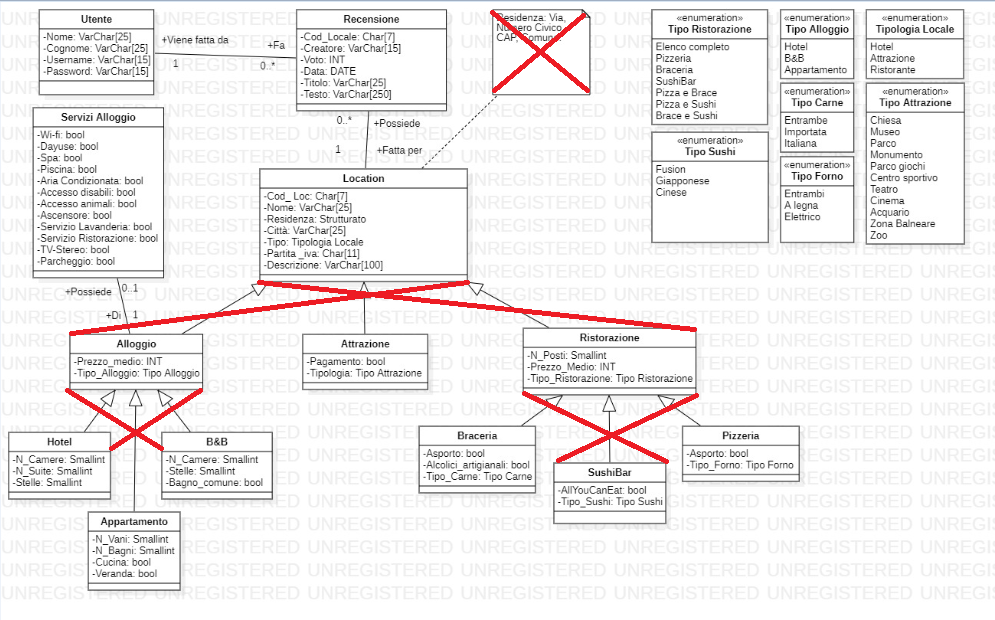
Infatti i vincoli che si sono introdotti (imposti anche dalla traccia) definiscono che la prima specializzazione è una total/disjoint ciò significa che un istanza “Location” può essere SOLO “Alloggio”, “Attrazione” o “Ristorante”.

Le successive specializzazioni invece si sono divise i 3 macro esempi:

* Alloggio: specializzazione partial/disjoint con “Hotel”, “BeB” e “Appartamento”
* Attrazione: per la vastità di tipologie si è pensato di non far avere specializzazioni da “Attrazione”
* Ristorante: specializzazione partial/overlapping con “Pizzeria”, “SushiBar” e “Braceria”

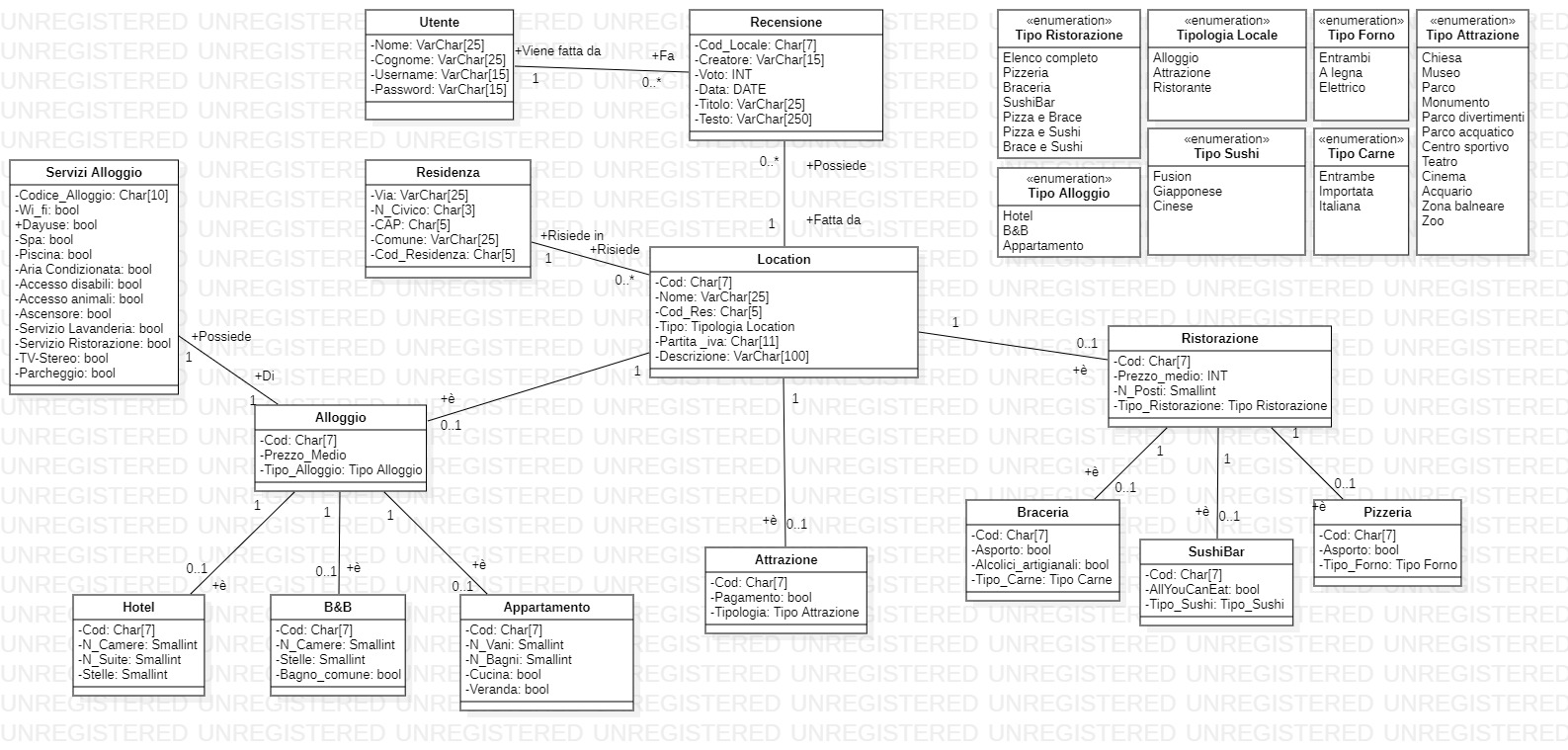
La ristrutturazione completa del class diagram ha portato (come c’era da aspettarsi) alla obbligatoria aggiunta di attributi di *chiave esterna* per istanziare le associazioni che hanno sostituito le specializzazioni

Infine per quanto riguarda la classe creatasi “Residenza”, gli attributi che la compongono son i campi strutturati precedentemente definiti, con l’aggiunta di una chiave surrogata per avere una *chiave primaria* a cui associarsi dalla classe “Location”.



Le “X” rosse rappresentano ciò che nel class diagram (ristrutturato) non apparirà.

**4 – Class diagram (Ristrutturato)**



Il class diagram sovrastante rappresenta la ricostruzioni adattabile alla traduzione dei dati logica e relazionale.

Come precedentemente illustrato il class diagram conteneva un attributo strutturato e varie specializzazione che andavano modellate per una più facile comprensione.

Ristrutturazione “Attributi strutturati”:

L’attributo strutturato “Residenza” contenuto nella tabella “Location” è stato prontamente eliminato per rappresentarlo sotto forma di classe a se stante.

La classe creatasi, chiamata appunto “Residenza”, contiene come attributi quelli che sono le informazioni principali di una residenza, introducendo la chiave primaria (per introdurre unicità e associazione ad altre classi), l’attributo è stato chiamato per convenzione “Cod\_Residenza” associato con la chiave esterna di “Location” rinominata “Cod\_Res”.

L’associazione creatasi tra “Residenza” e “Location” è una associazione 1 a 0…\*, questa perché definisce l’ovvietà dove una residenza può avere associate nessuna o molte location mentre una location può avere un'unica residenza associata.

Ristrutturazione “Molteplicità”:

Assenti durante la progettazione del problema.

Ristrutturazione “Generalizzazioni/Specializzazioni”:

La ristrutturazione delle ben 9 specializzazioni della classe “Location” sono state gestite mantenendo intatta la struttura “ad Albero” senza complicazioni di schiacciamento o vincoli che permettessero la creazione di istanze solo in specifici casi.

Sono stati introdotti attributi creati (gestiti con apposite enumerazioni) alcune introdotte per creare una piccola differenza tra e varie specializzazioni altre sono invece attributi “Surrogati” usati come switch per identificare la specializzazione successiva e facilitare il lavoro sull’applicativo JAVA.

Esempio pratico:

Tipo\_Location, identifica la tipologia tra Alloggio, Attrazione e Ristorante.

Identificata la tipologia di location abbiamo (a seconda della scelta) un altro attributo (definito nella classe specializzata da location) che identifica il Tipo che successivamente si andrà a specializzarsi (ad eccezione della classe “Attrazione”).

N.B

Le specializzazioni create sono per la classe “Alloggio” partial/disjoint mentre per la calsse “Ristorante” partial/overlapping, ovviamente sono parziali per il semplice fatto che ci sono altre istanze che possono essere considerate nel problema ma che per semplicità ne vengono rappresentate solo 3.

Disjoint sono le specializzazioni di alloggio dato che per naturalezza un “Hotel” non può essere un “BeB” oppure un “Appartamento

Overlapping dato che un “Ristorante” può essere sia “Pizzeria” che “Braceria” che “SushiBar” (si consideri il viceversa).

L’unica total/disjoint è quella definita dalla traccia ovvero la specializzazione “Alloggio”, “Attrazione” e “Ristorante” dato che rappresentano le tre istanze del problema completo.

**5 – Dizionario delle classi**

**6 – Dizionario delle associazioni**

**7 – Dizionario dei vincoli**

**8 – Schema logico/relazionale**

Lo schema relazionale per il problema rappresenterà tutte le relazioni con classi, chiavi primarie, chiavi esterne senza ordine ne molteplicità:

N.B i domini non specificati sono null o comunque non rilevanti.

Utente(Nome, Cognome, Username, Password);

Numero attributi:4

Chiavi primarie:1

Chiavi esterne:0

---

Recensione(Cod\_Locale, Creatore, Voto, Data, Titolo, Testo);

Numero attributi:6

Chiavi primarie:0

Chiavi esterne:2

Dom(Voto)={1…5}

Dom(Data)={Giorno=1…31, Mese=1..12, Anno=1900…2100}

---

Location(Cod, Nome, Cod\_Res, Tipo, Partita\_IVA, Descrizione);

Numero attributi:6

Chiavi primarie:1

Chiavi esterne:1

Dom(Tipo)={Alloggio, Attrazione, Ristorante}

---

Residenza(Via, N\_Civico, CAP, Comune, Cod\_Residenza);

Numero attributi:5

Chiavi primarie:1

Chiavi esterne:0

---

Alloggio(Cod, Prezzo\_medio, Tipo\_Alloggio);

Numero attributi:3

Chiavi primarie:0

Chiavi esterne:1

Dom(Tipo\_Alloggio)={Hotel, BeB, Appartamento}

---

Servizi Alloggio(Codice\_Alloggio, Wi\_Fi, Dayuse, Spa, Piscina, Aria Condizionata, Accesso disabili, Accesso animali, Ascensore, Servizio Lavanderia, Servizio Ristorazione, TV\_Stereo, Parcheggio);

Numero attributi:13

Chiavi primarie:0

Chiavi esterne:1

(I domini degli attributi al di fuori di “Codice\_Alloggio” sono boolean)

---

Hotel(Cod, N\_Camere, N\_Suite, Stelle);

Numero attributi:4

Chiavi primarie:0

Chiavi esterne:1

Dom(Stelle)={1…5}

---

BeB(Cod, N\_Camere, Stelle, Bagno\_comune);

Numero attributi:4

Chiavi primarie:0

Chiavi esterne:1

Dom(Stelle)={1…5}

Dom(Bagno\_comune)={si, no, ns}

---

Appartamento(Cod, N\_Vani, N\_Bagni, Cucina, Veranda);

Numero attributi:5

Chiavi primarie:0

Chiavi esterne:1

Dom(Cucina)={si, no, ns}

Dom(Veranda)={si, no, ns}

---

Attrazione(Cod, Pagamento, Tipologia);

Numero attributi:3

Chiavi primarie:0

Chiavi esterne:1

Dom(Pagamento)={si, no, ns}

Dom(Tipologia)={Chiesa, Museo, Parco, Monumento, Parco divertimenti, Parco acquatico, Centro sportivo, Teatro, Cinema, Acquario, Zona balneare, Zoo}

---

Ristorazione(Cod, Prezzo\_medio, N\_Posti, Tipo\_Ristorazione);

Numero attributi:4

Chiavi primarie:0

Chiavi esterne:1

Dom(Tipo\_Ristorazione)={Elenco completo, Braceria, SushiBar, Pizzeria, Pizza e Brace, Pizza e Sushi, Brace e Sushi}

---

Braceria(Cod, Asporto, Alcolici\_artigianali, Tipo\_Carne);

Numero attributi:4

Chiavi primarie:0

Chiavi esterne:1

Dom(Asporto)={si, no, ns}

Dom(Alcolici\_artigianali)={si, no, ns}

Dom(Tipo\_Carne)={Entrambe, Importata, Italiana}

---

SushiBar(Cod, AllYouCanEat, Tipo\_Sushi);

Numero attributi:3

Chiavi primarie:0

Chiavi esterne:1

Dom(AllYouCanEat)={si, no, ns}

Dom(Tipo\_Sushi)={Fusion, Giapponse, Cinese}

---

Pizzeria(Cod, Asporto, Tipo\_Forno);

Numero attributi:3

Chiavi primarie:0

Chiavi esterne:1

Dom(Asporto)={si, no, ns}

Dom(Tipo\_Forno)={Entrambi, Elettrico, A legna}

**9 – SQL**