

**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

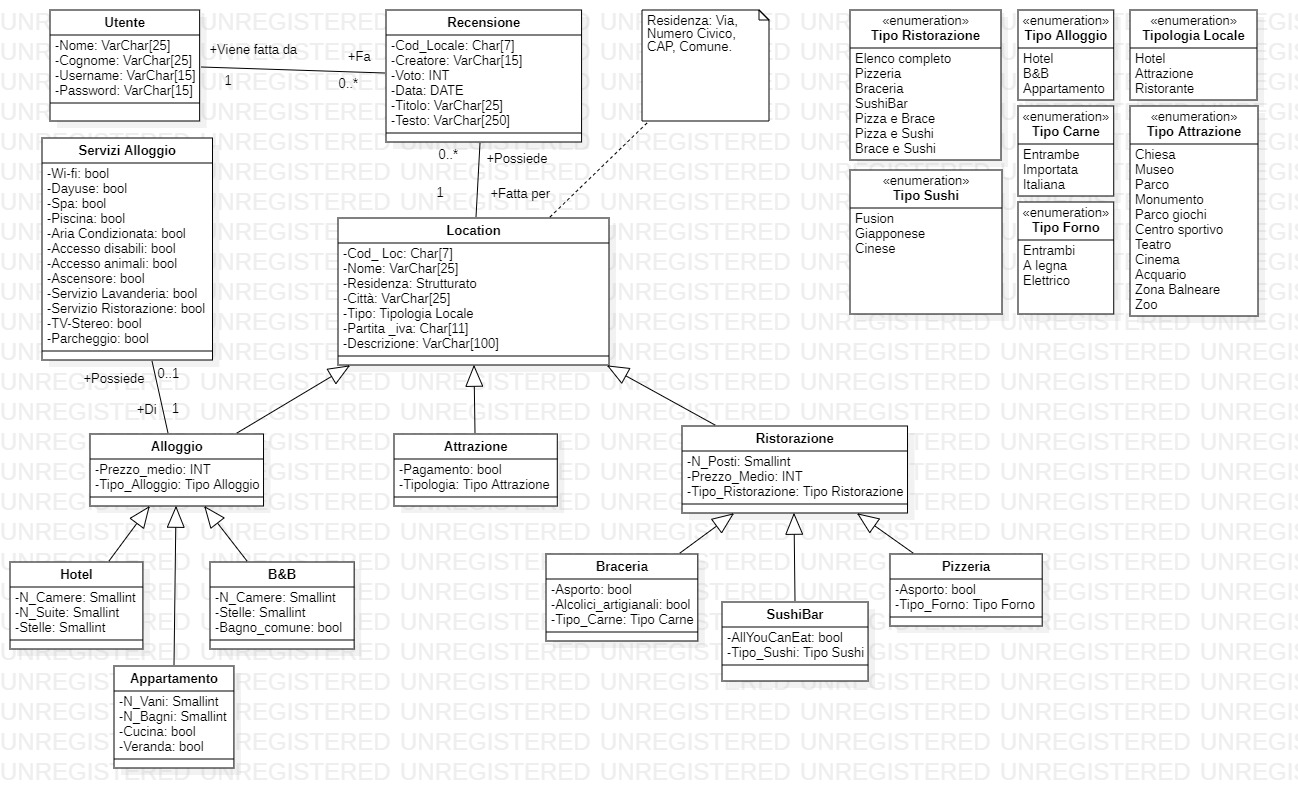
(N.B le specializzazioni devono essere  ulteriormente  raffinate)

Si è pensati 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

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 enumeration 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.

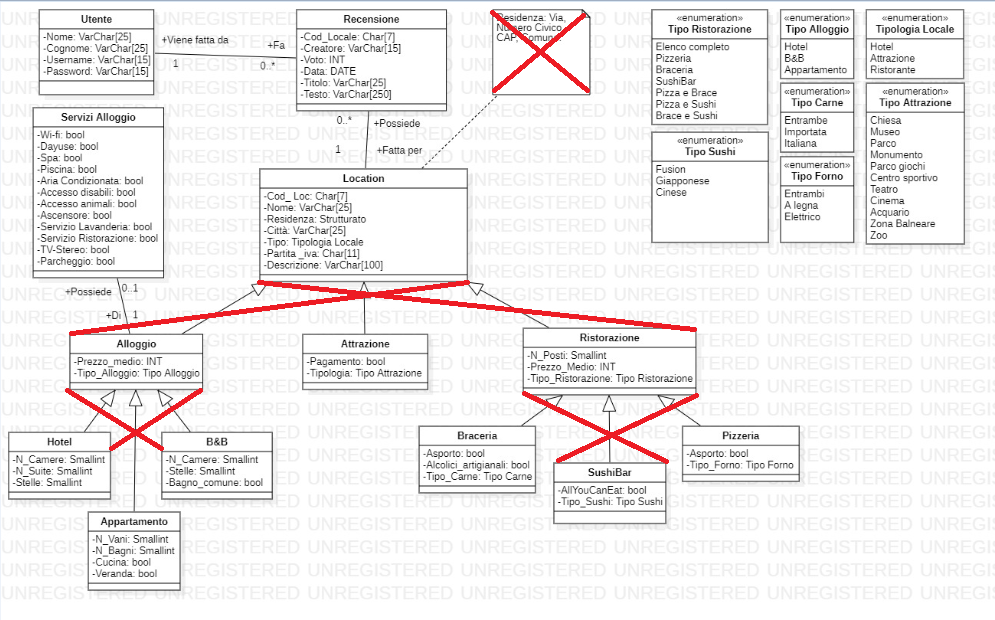
Infatti i vincoli che si sono introdotti (imposti anche dalla traccia) definiscono che la prima specializzazione è una total/disjoint ciò significa che un istanza “Location” può essere SOLO “Alloggio”, “Attrazione” o “Ristorante”.

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 foreign key 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 primary key a cui associarsi dalla classe “Location”.



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