PROGETTO **NomadListDB**

AUTORI **Paris , Riggi**

Corso di laurea: Informatica

Sommario

1. Parte Prima: Generalità

1.1. Descrizione generale del prodotto

Il nostro progetto si basa sul seguente sito web: https://nomadlist.com/.

Questo sito è un portale attraverso il quale i nomadi digitali raccolgono informazioni e scambiano pareri su possibili luoghi dove possono lavorare da remoto.

Un nomade digitale è colui che lavora a distanza per i propri clienti o per il proprio datore di lavoro, senza essere legato a sedi fisiche, così facendo, può lavorare in remoto, stare sempre in viaggio, conducendo una vita, per l'appunto, nomade. L'obiettivo di questo sito si sposa perfettamente con la moderna figura dell'informatico che può tranquillamente svolgere il suo lavoro da remoto.

L'obiettivo del nostro progetto è quello di costruire un database prendendo spunto dai dati che utilizza questo sito.

1.2. Obiettivi del Progetto:

Il nostro obiettivo, quindi, è quello di creare un Database in cui si trovano i dati necessari per il sito web, nello specifico:

- Memorizzare le città con i loro attributi:
  - Attributi principali: nome, stato di appartenenza e coordinate.
  - Attributi secondari: possiamo dividerli in generale e meteo.
    - Attributi generali: numero di abitanti, costo medio della vita, velocità della connessione misurata in Mbps, livello di sicurezza percepito, stipendio medio annuale in dollari, costo connessione mensile e family score.
    - Attributi meteo: data, temperatura in gradi Celsius, percentuale di umidità, percentuale di nuvolosità e indice di qualità dell'aria.

- Memorizzare gli utenti con i loro attributi: email, username e password.
  - Nota: Email e username sono univoci.

- Memorizzare quali lingue sono parlate nelle città. Questo può essere utile agli utenti che desiderano lavorare in un ambiente multilingue o che hanno bisogno di comunicare in una lingua specifica.

- Memorizzare i commenti che un utente può lasciare riguardo una città tenendo conto di data della scrittura e testo del commento.

- Memorizzare i mi piace che un utente può mettere, tenendo conto che gli utenti possono mettere un solo mi piace per ogni città e/o un solo mi piace per ogni commento di altri utenti.

- Memorizzare quali città ha visitato ogni utente e in che data, tenendo conto che un utente può non aver visitato nessuna città o averne visitata più di una.

1.3. Utenti

Il prodotto è destinato principalmente ai nomadi digitali che tramite i nostri dati possono raccogliere informazioni sui luoghi dove poter lavorare.

Elenco delle categorie di utenza (ruoli), profilo di ciascuna categoria e obiettivi/bisogni in rapporto al progetto:

- Amministratore della base di dati: proprietario del sito.
- Progettisti e programmatori: devono progettare e implementare la base di dati, devono quindi conoscere le modalità di progettazione e devono conoscere le tecnologie utili all'implementazione e alle eventuali modifiche e aggiornamenti.
- Utenti finali: sono persone che accedono al sito, principalmente nomadi digitali in cerca di informazioni per lavorare in città sempre nuove.

2. Parte seconda: Raccolta e analisi dei Requisiti

Analisi dell'utente: il progetto è destinato ai nomadi digitali.

Analisi dei bisogni: gli utenti che si approcciano al nostro progetto hanno bisogno di:

1. Conoscenza approfondita delle città che viene fornita tramite i dati memorizzati, che vanno dallo stipendio medio annuo alla temperatura passando per tanti altri.
2. Sapere cosa ne pensano gli altri utenti riguardo le città a cui sono interessati.

2.1. Realizzazioni preesistenti

Il nostro progetto prende ispirazione dai dati che il sito https://nomadlist.com/ utilizza, sui quali ci siamo basati senza cercare di replicarli perfettamente ma dando una nostra interpretazione.

2.2. Elenco dei requisiti

- Requisiti di carattere generale:
  - Si vuole realizzare un database basato sul sito nomadlist.com, del quale vogliamo rappresentare gli utenti, le città e le relazioni che ci sono tra loro.

- Requisiti relativi alle città:
  - Per le città identificate dalle loro coordinate vogliamo rappresentare il nome e lo stato di appartenenza, le loro informazioni suddivise in informazioni generali e informazioni meteo, rappresentiamo anche i commenti e i like che gli utenti lasciano alle città.

- Requisiti relativi all'utente:

- Per gli utenti identificati dalla loro email, vogliamo rappresentare il loro nome utente e la password, l'utente può lasciare un commento relativo ad una città e un like come apprezzamento ad altri commenti o a città stesse.

- Requisiti relativi alle visite:
  - Vogliamo rappresentare le visite che gli utenti fanno alle città, le identifichiamo tramite un id e vogliamo rappresentare una data di inizio e di fine del viaggio.

- Requisiti relativi alle informazioni:
  - Per le informazioni identificate da un id distinguiamo informazioni generali e informazioni meteo
    - Per le informazioni che sono di tipo generale rappresentiamo la sicurezza della città (valore da 1 a 5), lo stipendio medio annuo (in dollari) , il numero di abitanti, il costo medio di vita mensile(in dollari), la velocità di connessione media (in Mbps per secondo), family score (valore da 1 a 5) e costo connessione mensile (in dollari). Per le informazioni di tipo meteo rappresentiamo la data di previsioni meteo a cui fanno riferimento i dati, temperatura (misurata in gradi Celsius), la percentuale di umidità, la percentuale di nuvolosità e la qualità dell'aria.

- Requisiti relativi ai commenti:
  - Per i commenti identificati da un id vogliamo rappresentare il testo e la data di scrittura, vengono lasciate da un utente e commentano una specifica città, possono ricevere like dagli utenti.

- Requisiti relativi alle lingue:
  - Ogni lingua ha un proprio nome che la identifica

2.3. Glossario dei termini

Termine	**Descrizione**	**Sinonimi**	**Collegamenti**
Città	Città presenti nel database.	Comune, Centro abitato	Visita,utente, commento,info
Informazioni	Informazioni riguardo la città, possono essere di carattere generale o meteo.	Dati	Città
Utente	Persona che si registra al sito	Utilizzatore	Commento,città, visita
Visite	Visita di una città da parte di un utente	Viaggio	Utente, città
Costo connessione	Per costo connessione si intende il costo mensile medio che bisogna pagare per avere accesso ad una	dati mobili	Info generali

	connessione dati mobili di 10 Gb		
Family score	Con family score si intende un intero che va da 1 a 5 che rappresenta la qualità di vita per le famiglie	vivibilità per famiglie	Info generali
Sicurezza	Con sicurezza si intende il valore percepito di sicurezza della città.	Salvaguardia	Info generali

2.4. Specifiche, assunzioni e vincoli d'integrità

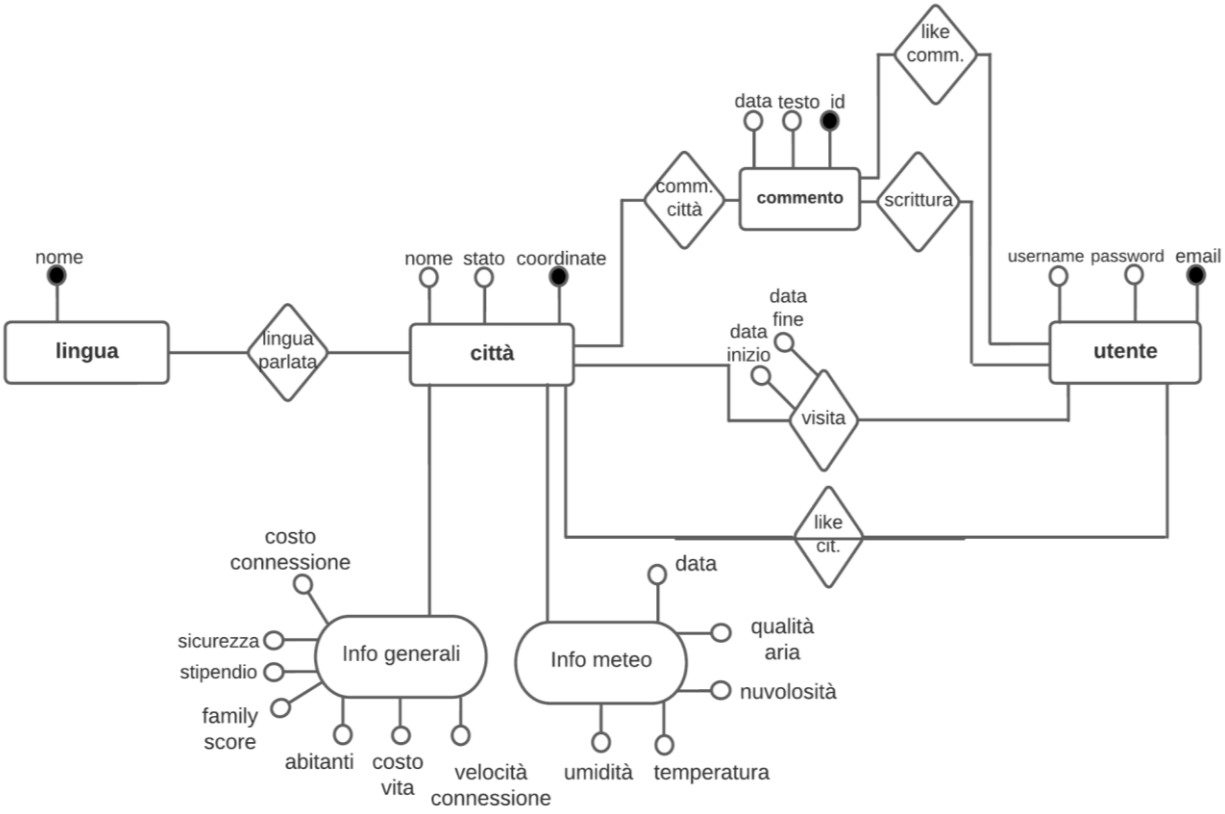
- **Assunzioni:**
  - **A1:** Attributi riferiti alle città quali nome, stato di appartenenza, coordinate e numero di abitanti vengono prese da API esterne.
  - **A2:** Le informazioni riguardanti i dati meteo delle città vengono prese da API esterne e aggiornate con update lato backend 3 volte al giorno.

- **Vincoli di integrità:**
  - **V1:** Un utente non può visitare più di una volta la stessa città nella stessa data
  - **V2:** Il nome di ogni città deve corrispondere al nome di una città esistente
  - **V3:** Lo stato della città deve corrispondere al nome di uno stato esistente e deve essere lo stato a cui appartiene la città stessa (es. se città = Roma allora stato = Italia)
  - **V4:** La data di fine viaggio non può essere precedente alla data di inizio viaggio.
  - **V5:** La data di inizio o di fine viaggio non può essere successiva alla data odierna.
  - **V6:** I dati di umidità e nuvolosità devono essere compresi tra 0 e 100 essendo percentuali.
  - **V7:** I dati sul livello di sicurezza vanno da 1 (meno sicuro) a 5 (più sicuro).
  - **V8:** Un commento può avere massimo 250 caratteri.
  - **V9:** Due utenti non possono avere lo stesso nome utente.
  - **V10:** Un utente non può lasciare più di un like ad una stessa città o ad uno stesso commento.
  - **V11:** La data di scrittura di un commento non può essere successiva alla data odierna.
  - **V12:** Un utente può registrare una visita verso una città solo quando essa è terminata.
  - **V13**: Un utente non può effettuare più di una visita nello stesso arco temporale. Vale a dire che dato un utente u, data una visita v con i data di inizio e f data di fine, siano v' una nuova visita da inserire con i' E f' relative date di inizio e fine, allora non possiamo avere : (i <= i' AND f >= i') OR (i <= f' AND f >= f') ; cioè a parole significa che non si può avere che v' inizi o finisca mentre è in corso v.

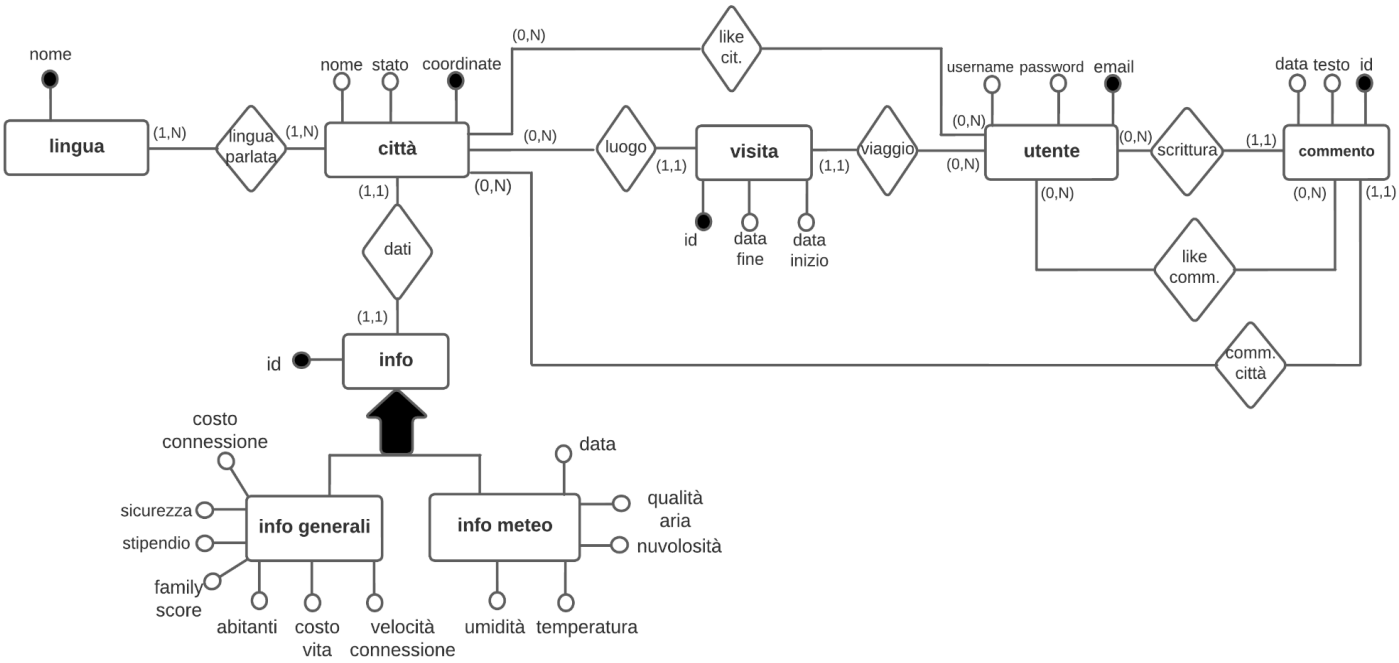
3. Parte Terza: Progettazione concettuale

3.1. Diagramma E-R

- **Schema scheletro**



- **Raffinamenti successivi (ER finale)**



3.2. Dizionario dei Dati

Entità	Attributi	Chiave	Descrizione
Città	coordinate:stringa nome:stringa stato:stringa	coordinate	Ogni istanza rappresenta una città specifica e contiene le coordinate ,un nome e lo stato di appartenenza
Utente	email:stringa username:stringa password:stringa	email	ogni istanza rappresenta una persona di cui vengono memorizzati uno username, l'email e la password
Commento	id:numero intero>0 testo:stringa data:data	id	ogni istanza rappresenta un commento che un utente scrive per una

			città in una determinata data
Visita	id:numero intero>0 data_inizio:data data_fine:data	id	ogni istanza rappresenta la visita di una città da parte di un utente svolta in un lasso di tempo che va da data_inizio a data_fine
Info	id:numero intero>0	id	entità padre, ogni istanza rappresenta le informazioni associate ad un'istanza di città.
Info generali	id:numero intero>0 abitanti: numero intero>0 velocità connessione: numero intero>0 costo vita: numero intero>0 stipendio:numero intero>0 sicurezza: numero intero da 1 a 5 family score: numero intero da 1 a 5 costo connessione: intero>0	id	entità figlia che eredita la chiave dall'entità padre "Info", ogni istanza contiene informazioni generali riguardanti una città
Info meteo	id: numero intero>0 data: data nuvolosità: numero intero compreso tra 0 e 100 qualità aria: numero intero>=0 umidità: numero intero compreso tra 0 e 100 temperatura: numero intero	id	entità figlia che eredita la chiave dall'entità padre "Info", ogni istanza contiene informazioni meteo riguardanti una città
Lingua	nome: stringa	nome	Ogni istanza dell'entità Lingua rappresenta una

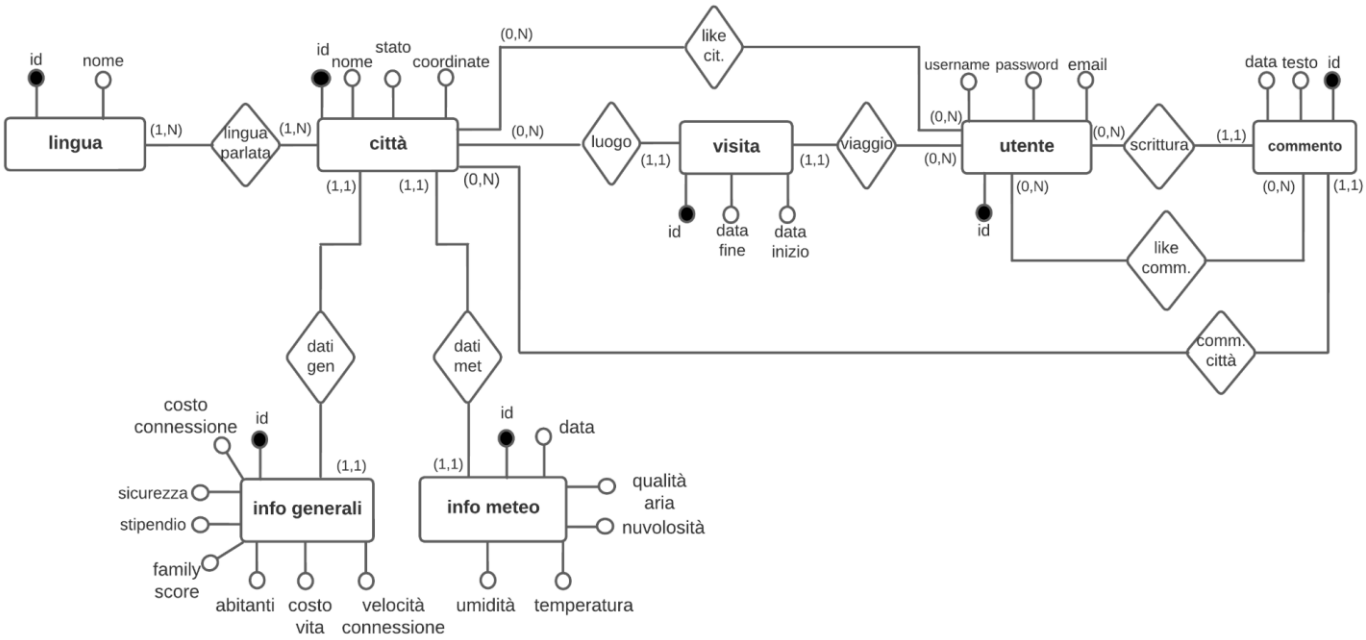
			lingua parlata in qualche città

Relazione	Cardinalità	Descrizione
Dati	Info(1,1); Citta(1,1)	Associa un'informazione alla città. Un'istanza di città partecipa a minimo 1 massimo 1 occorrenze di "Dati" con l'entità Info; un'istanza di Info partecipa a minimo 1 massimo 1 occorrenze di "Dati" con l'entità Citta.
Luogo	Visita(1,1); Citta(0,N)	Le città sono luogo di visita da parte degli utenti. Un'istanza di Città partecipa a minimo 0 massimo N occorrenze di "Luogo" con l'entità Visita; un'istanza di Visita partecipa a minimo 1 massimo 1 occorrenze di "Luogo" con l'entità Citta
Viaggio	Visita(1,1);Utente(0,N)	Gli utenti possono aver viaggiato in alcune città. Un'istanza di utente partecipa a minimo 0 massimo N occorrenze di "Viaggio" con l'entità Visita; Un'istanza di Visita partecipa a minimo 1 massimo 1 occorrenze di "Viaggio" con l'entità Utente.
Scrittura	Commento(1,1);Utente(0,N)	Gli utenti possono scrivere un commento. Un'istanza di Utente partecipa a minimo 0 massimo N occorrenze di "Scrittura" con l'entità Commento; un'istanza di Commento partecipa a minimo 1 massimo 1 occorrenze di "Scrittura" con l'entità Utente
Comm città	Commento(1,1);Città(0,N)	Le città possono ricevere commenti. Un'istanza di commento partecipa a minimo 1 massimo 1 occorrenze di "Comm citta" con l'entità Città; un'istanza di Città partecipa minimo a 0 massimo N occorrenze di "Comm città" con l'entità Commento

Like cit	Citta(0,N); Utente(0,N)	Gli utenti possono mettere like alle città. Un'istanza di Utente partecipa a minimo 0 massimo N occorrenze di "Like cit" con l'entità Citta; un'istanza di Citta partecipa a minimo 0 massimo N occorrenze di "Like cit" con l'entità Utente.
Like comm	Commento(0,N), Utente(0,N)	Gli utenti possono mettere like ai commenti. Un'istanza di Utente partecipa a minimo 0 massimo N occorrenze di "Like comm" con l'entità Commento; un'istanza di Commento partecipa a minimo 0 massimo N occorrenze di "Like comm" con l'entità Utente.
Lingua parlata	Citta(1,N) , Lingua(1,N)	In ogni città si possono parlare 1 o più lingue e ogni lingua può essere parlata in 1 o più città. Un'istanza di città partecipa a minimo 1 massimo N occorrenze di "Lingua parlata" con l'entità lingua; un'istanza di Lingua partecipa a minimo 1 massimo N occorrenze di "Lingua Parlata" con l'entità Citta.

4. Parte Quarta: Progettazione Logica

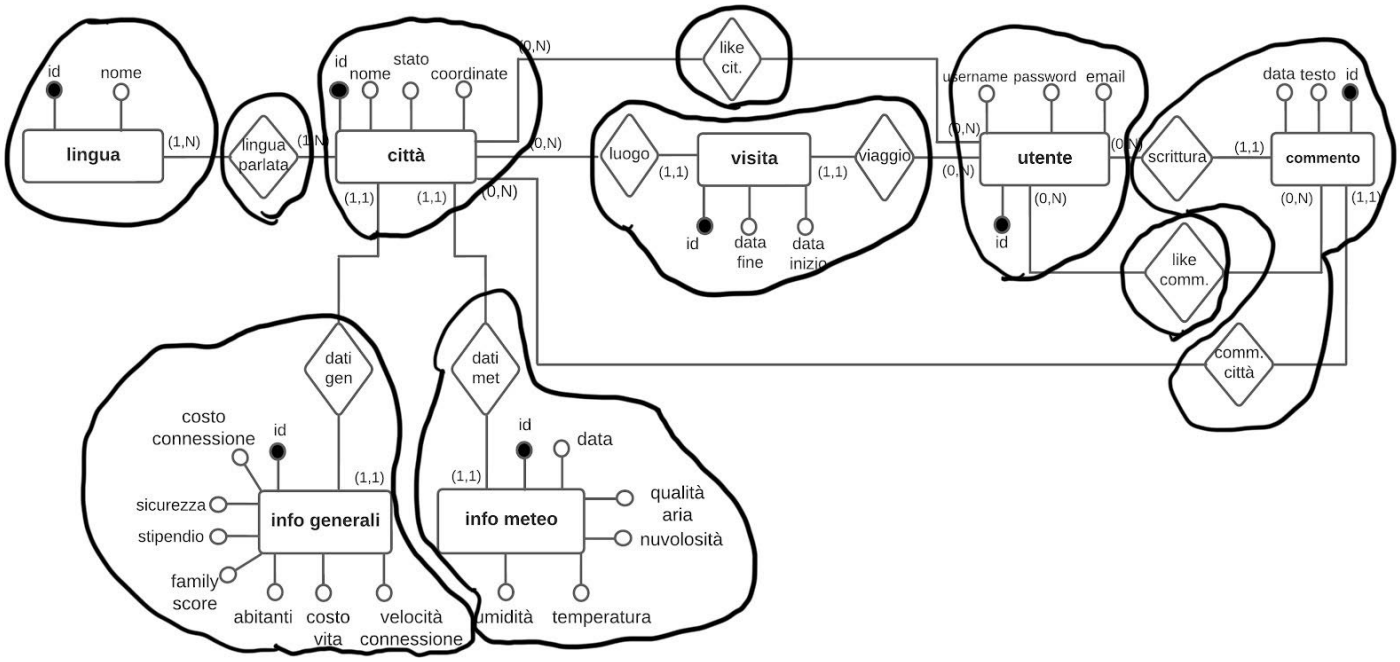
4.1. Schema E-R concettuale ristrutturato



Abbiamo eliminato la generalizzazione accorpando l'entità genitore (info) nelle entità figlie (info generali e info meteo). Quest'ultime ereditano l'id e la relazione che legava info a citta. Nel nostro caso questa scelta è la più conveniente in quanto l'entità padre ha solo un attributo e una relazione, e ci permette sia di risparmiare memoria non avendo mai valori null sia di ridurre gli accessi.

Nell'ER ristrutturato abbiamo preferito sostituire le chiavi di formato stringa con id di tipo intero. Questa scelta è stata fatta per ricercare una maggiore efficienza (ad esempio nelle operazioni di confronto e unione), per avere prestazioni maggiori (per esempio durante il join dove operazioni tra tabelle basate su chiavi intere sono più veloci rispetto alle operazioni basate su stringhe) e per ridurre lo spazio occupato (non dovendo memorizzare ogni volta che c'è un riferimento alla chiave di tipo varchar tutti i caratteri che contiene, ma piuttosto un semplice intero).

4.2. Passaggio da schema E-R ristrutturato a logico



4.3. Schema E-R logico

Utente (id, email, username, password)

Citta (id,nome,stato,coordinate)

Info_meteo(id,data,temperatura,umidità,nuvolosita,qualita_aria,citta_id)

Foreign key (citta_id) references Citta(id)

Info_generali(id,abitanti,costo_connessione,costo_vita,sicurezza,stipendio,family_score ,velocita_connessione,citta_id)

Foreign key (citta_id) references Citta(id)

Commento (id,testo,data,utente_id,citta_id)

Foreign key (citta_id) references Citta(id)

Foreign key (utente_id) references Utente(id)

Visita (id,data_inizio,data_fine,citta_id,utente_id)

Foreign key (citta_id) references Citta(id)

Foreign key (utente_id) references Utente(id)

Like_cit (citta_id,utente_id)

Foreign key (citta_id) references Citta(id)

Foreign key (utente_id) references Utente(id)

Like_comm(commento_id,utente_id)

Foreign key (commento_id) references Commento(id)

Foreign key (utente_id) references Utente(id)

Lingua(id,nome)

Lingua_parlata(lingua_id,citta_id)

Foreign key (lingua_id) references Lingua(id)

Foreign key (citta_id) references Citta(id)

4.4. Dizionario

UTENTE			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
id	int	4 byte	Primary key Unsigned Auto_increment
email	varchar(50)	51 byte	Unique Not null
username	varchar(20)	21 byte	Unique Not null
password	Varchar(40)	41 byte	Not null

CITTA			
Attributo	**Tipo**	**Dimensione**	**Vincoli**

id	int	4 byte	Primary key Unsigned Auto_increment
nome	Varchar(30)	31 byte	Not null
stato	Varchar(30)	31 byte	Not null
coordinate	varchar(70)	71 byte	Unique Not null

COMMENTO			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
id	int	4 byte	Primary key Unsigned Auto_increment
testo	Varchar(250)	251 byte	Not null
data	Timestamp	8 byte	Not null
citta_id	int	4 byte	Unsigned Foreign key (citta_id) references città(id)
utente_id	int	4 byte	Unsigned Foreign key (utente_id) references utente(id)

INFO_GENERALI

Attributo	Tipo	Dimensione	Vincoli
id	int	4 byte	Primary key Unsigned Auto_increment
sicurezza	int	4 byte	Not null Check (sicurezza>=1 and sicurezza<=5)
abitanti	int	4 byte	Unsigned Not null
costo_connessione	Int	4 byte	Unsigned Not null
costo_vita	int	4 byte	Unsigned Not null
stipendio	int	4 byte	Unsigned Not null
family_score	int	4 byte	Not null Check (sicurezza>=1 and sicurezza<=5)
velocita_connessione	int	4 byte	Unsigned not null
citta_id	int	4 byte	Unsigned unique Foreign key(citta_id) references citta(id)

INFO_METEO			
Attributo	**Tipo**	**Dimensione**	**Vincoli**

id	int	4 byte	Primary key Unsigned Auto_increment
data	Date	3 byte	Not null
temperatura	int	4 byte	Not null Check (temperatura>-100 and temperatura<100)
umidita	int	4 byte	Unsigned Check(umidita<=100) Not null
nuvolosita	int	4 byte	Unsigned Check(nuvolosita<=100) Not null
qualita_aria	int	4 byte	Unsigned Not null
citta_id	int	4 byte	Unsigned unique Foreign key(citta_id) references citta(id)

VISITA			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
id	int	4	Primary key Unsigned Auto_increment
data_inizio	Date	3	Not null

data_fine	Date	3	Not null
citta_id	int	4	Unsigned Foreign key(citta_id) references citta(id)
utente_id	int	4	Unsigned Foreign key(utente_id) references utente(id)
			Unique(utente_id, data_inizio, data_fine)

LIKE_CIT			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
citta_id	int	4	Unsigned Foreign key(citta_id) references citta(id)
utente_id	int	4	Unsigned Foreign key(utente_id) references utente(id)
			Primary key(citta_id, utente_id)

LIKE_COMM			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
commento_id	int	4	Unsigned Foreign key(commento_id) references commento(id)

utente_id	int	4	Unsigned Foreign key(utente_id) references utente(id)
			Primary key(commento_id, utente_id)

LINGUA			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
id	int	4	Primary key Unsigned Auto_increment
nome	Varchar(20)	21	Unique

LINGUA_PARLATA			
Attributo	**Tipo**	**Dimensione**	**Vincoli**
lingua_id	int	4	Unsigned Foreign key(lingua_id) references lingua(id)
citta_id	int	4	Unsigned Foreign key(citta_id) references citta(id)
			Primary key(lingua_id, utentecitta_id_id)

4.5. Indici di prestazione e carico applicativo

- **Operazioni più frequenti:**

  Operazione 1: Inserimento di una città

  Operazione 2: Inserimento di un utente

  Operazione 3: Inserimento di un commento

  Operazione 4: Inserimento di una visita

  Operazione 5: Inserimento info generali di una città

  Operazione 6: Inserimento info meteo di una città

  Operazione 7: Inserimento di un like da parte degli utenti alle città

  Operazione 8: Inserimento di un like da parte degli utenti ai commenti

  Operazione 9: Lettura di una singola città

  Operazione 10: Lettura info generali relative a una città

  Operazione 11: Lettura info meteo relative a una città

  Operazione 12: Lettura numero like relativi ad una città

  Operazione 13: Lettura di tutte le città

  Operazione 14: Lettura di tutti i commenti di una città

  Operazione 15: Lettura del numero di like di un commento

  Operazione 16: Aggiornamento dei dati meteo di una città

- **Tavola delle operazioni**

Operazione	Tipo	Frequenza (al giorno)
Op.1	B	Nel primo inserimento circa 100, successivamente al massimo una decina al giorno.
Op.2	I	15
Op.3	I	30
Op.4	I	5
Op.5	B	#Op.5 = #Op.1
Op.6	B	#Op.6 = #Op.1
Op.7	I	45
Op.8	I	25
Op.9	I	250

Operazione	Tipo	Frequenza (al giorno)
Op.10	I	100
Op.11	I	100
Op.12	I	100
Op.13	I	150
Op.14	I	70
Op.15	I	70
Op.16	B	3

4.6. Volume dei dati

Relazione	Occorrenze	Grandezza
Citta	600	82.200 byte
Info_generali	600	21.600 byte
Info_meteo	600	16.200 byte
Visita	8000	144.000 byte
Utente	4000	468.000 byte
Commento	1.500.000	406.500.000 byte
Like_cit	35.000	280.000 byte
Like_comm	700.000	5.600.000 byte
Lingua	400	10.000 byte
Lingua_parlata	1000	8.000 byte

Somma totale grandezza delle occorrenze: 413.130.000 byte

5. Parte Quinta: Progettazione Fisica

Schema fisico - creazione database e tabelle

```
CREATE SCHEMA NOMADLIST_DB;
USE NOMADLIST_DB;

CREATE TABLE utente (
        id int UNSIGNED AUTO_INCREMENT,
        email varchar(50) UNIQUE NOT NULL,
        username varchar(20) UNIQUE NOT NULL,
        password varchar(40) NOT NULL,
        PRIMARY KEY(id)
);

CREATE TABLE citta(
        id int UNSIGNED AUTO_INCREMENT,
        nome varchar(30) NOT NULL,
        stato varchar(30) NOT NULL,
        coordinate varchar(70) UNIQUE NOT NULL,
        PRIMARY KEY(id)
);

CREATE TABLE commento(
        id int UNSIGNED AUTO_INCREMENT,
        testo varchar(250) NOT NULL,
        data timestamp NOT NULL,
        citta_id int UNSIGNED,
        utente_id int UNSIGNED,
        PRIMARY KEY(id),
        FOREIGN KEY (citta_id) REFERENCES citta(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        FOREIGN KEY (utente_id) REFERENCES utente(id)
        ON UPDATE CASCADE
        ON DELETE CASCADE
);
```

```
CREATE TABLE info_generali(
        id INT UNSIGNED AUTO_INCREMENT,
        sicurezza int NOT NULL CHECK (sicurezza>=1 and sicurezza<=5),
        abitanti int UNSIGNED NOT NULL,
        costo_connessione int UNSIGNED NOT NULL,
        costo_vita int UNSIGNED NOT NULL,
        stipendio int UNSIGNED NOT NULL,
        family_score int NOT NULL CHECK (family_score>=1 and family_score<=5),
        velocita_connessione int UNSIGNED NOT NULL,
        citta_id int UNSIGNED UNIQUE,
        PRIMARY KEY(id),
        FOREIGN KEY (citta_id) REFERENCES citta(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE info_meteo(
        id INT UNSIGNED AUTO_INCREMENT,
        data DATE NOT NULL,
        temperatura int NOT NULL CHECK(temperatura>-100 and temperatura<100),
        umidita int UNSIGNED NOT NULL CHECK( umidita<=100),
        nuvolosita int UNSIGNED NOT NULL CHECK( nuvolosita<=100 ),
        qualita_aria int UNSIGNED NOT NULL,
        citta_id int UNSIGNED UNIQUE,
        PRIMARY KEY(id),
        FOREIGN KEY (citta_id) REFERENCES citta(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE visita(
        id int UNSIGNED AUTO_INCREMENT,
        data_inizio date NOT NULL,
        data_fine date NOT NULL,
        citta_id INT UNSIGNED,
        utente_id INT UNSIGNED,
        PRIMARY KEY( id ),
        UNIQUE(utente_id,data_inizio,data_fine),
        FOREIGN KEY (citta_id) REFERENCES citta(id)
```

```
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        FOREIGN KEY (utente_id) REFERENCES utente(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE like_cit(
        citta_id INT UNSIGNED,
        utente_id INT UNSIGNED,
        PRIMARY KEY( citta_id , utente_id ),
        FOREIGN KEY (citta_id) REFERENCES citta(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        FOREIGN KEY (utente_id) REFERENCES utente(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE like_comm(
        commento_id INT UNSIGNED,
        utente_id INT UNSIGNED,
        PRIMARY KEY( commento_id , utente_id ),
        FOREIGN KEY (commento_id) REFERENCES commento(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        FOREIGN KEY (utente_id) REFERENCES utente(id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE lingua(
        id int UNSIGNED AUTO_INCREMENT,
        nome varchar(20) UNIQUE,
        PRIMARY KEY (id)
);

CREATE TABLE lingua_parlata(
        lingua_id int UNSIGNED,
```

```
    citta_id int UNSIGNED,
    PRIMARY KEY (lingua_id,citta_id),
    FOREIGN KEY (lingua_id) REFERENCES lingua(id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
    FOREIGN KEY (citta_id) REFERENCES citta(id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```

5.1. Trigger

Abbiamo progettato questo trigger per implementare il vincolo **v13**

```
DELIMITER //
CREATE TRIGGER controlla_visita_duplicata BEFORE INSERT ON visita
FOR EACH ROW
BEGIN
   IF EXISTS (
        SELECT 1
        FROM visita
        WHERE utente_id = NEW.utente_id AND (data_inizio <= NEW.data_inizio AND
data_fine >= NEW.data_inizio OR data_inizio <= NEW.data_fine AND data_fine >=
NEW.data_fine)
   ) THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Un utente non può fare più
visite nello stesso arco temporale';
   END IF;
END //
delimiter ;
```

Quando si utilizza EXISTS, non è importante quale valore specifico venga selezionato, poiché il risultato della clausola EXISTS dipende solo dal fatto che la query restituisca almeno una riga o nessuna. Abbiamo quindi scelto di mettere 1 al SELECT in quanto in questi casi si usa per convenzione.

La clausola `SIGNAL SQLSTATE '45000'` viene utilizzata per generare un'eccezione personalizzata in MySQL, quindi l'abbiamo utilizzata per scrivere un nostro messaggio personalizzato.

5.2. Inserimenti

```
INSERT INTO citta (nome,stato,coordinate) VALUES ("London","United Kingdom","latitude: 51.5085,longitude: -0.1257");

INSERT INTO citta (nome,stato,coordinate) VALUES ("Roma","Italia","latitude: 41.8933203,longitude: 12.4829321");

INSERT INTO citta (nome,stato,coordinate) VALUES ("Madeira","Portugal","latitude: 32.751750099999995,longitude: -16.98174865726062");

INSERT INTO citta (nome,stato,coordinate) VALUES ("Melbourne","Australia","latitude: -37.8142176,longitude: 144.9631608");

INSERT INTO citta (nome,stato,coordinate) VALUES ("Istanbul","Türkiye","latitude: 41.0091982,longitude: 28.9662187");

INSERT INTO citta (nome,stato,coordinate) VALUES ("Barcelona","Spain","latitude: 41.3828939,longitude: 2.1774322");

INSERT INTO citta (nome, stato, coordinate) VALUES ('Paris', 'France', 'latitude: 48.8566, longitude: 2.3522');

INSERT INTO citta (nome, stato, coordinate) VALUES ('Tokyo', 'Japan', 'latitude: 35.6895, longitude: 139.6917');

INSERT INTO citta (nome, stato, coordinate) VALUES ('Cairo', 'Egypt', 'latitude: 30.0444, longitude: 31.2357');

INSERT INTO citta (nome, stato, coordinate) VALUES ('Rio de Janeiro', 'Brazil', 'latitude: -22.9068, longitude: -43.1729');

INSERT INTO citta (nome, stato, coordinate) VALUES ('Sydney', 'Australia', 'latitude: -33.8688, longitude: 151.2093');

INSERT INTO citta (nome, stato, coordinate) VALUES ('Firenze', 'Italia', 'latitude: 43.769562, longitude: 11.255814');


INSERT INTO utente (email,username,password) VALUES ("jasonstatham@gmail.com","Jason_Statham","5f4dcc3b5aa765d61d8327deb882cf99") ;

INSERT INTO utente (email,username,password) VALUES ("paolobonolis@gmail.com","Paolo_bonolis","57496b1c3bae5a66e31e402a7487d77a");

INSERT INTO utente (email,username,password) VALUES ("pablopicasso@gmail.com","PicassoPablo","a1bf478497ca7ed8e0301c64ad01d813");

INSERT INTO utente (email,username,password) VALUES ("mattiapascal@gmail.com","IlFuMattiaPascal","8634a95f514364265e94b4176d6417b0 ");

INSERT INTO utente (email,username,password) VALUES ("queen_elizabeth@gmail.com","Regina","c9f1d3ad78bad371cc589d2f1d466e5a");

INSERT INTO utente (email,username,password) VALUES ("nino.frassica@gmail.com","Frassica","c9f1d3ad78bad371cc589d2f1d466e5a");

INSERT INTO utente (email,username,password) VALUES ("valerio.lundini@hotmail.it","Lundini","b2f6bb43ae8d325a42999d82e7c8cac7");

INSERT INTO utente (email, username, password) VALUES ('mario.rossi@gmail.com', 'MarioRossi', '5f4dcc3b5aa765d61d8327deb882cf99');
```

INSERT INTO utente (email, username, password) VALUES ('laura.bianchi@gmail.com', 'LauraBianchi', '202cb962ac59075b964b07152d234b70');

INSERT INTO utente (email, username, password) VALUES ('giuseppe.verdi@gmail.com', 'GiuseppeVerdi', 'e10adc3949ba59abbe56e057f20f883e');

INSERT INTO visita (data_inizio,data_fine,citta_id,utente_id) VALUES ("1930-03-12","2000-01-26",1,5);

INSERT INTO visita (data_inizio,data_fine,citta_id,utente_id) VALUES ("2018-06-16","2018-07-10",6,7);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ("2022-09-10", "2022-11-20", 3, 2);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ("2023-05-01", "2023-08-10", 4, 1);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ("2023-01-15", "2023-06-30", 6, 5);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ('2018-09-01', '2018-09-15', 5, 4);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ('2017-06-16', '2017-09-30', 3, 7);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ('2000-07-05', '2000-08-20', 2, 8);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ('2023-08-01', '2023-08-15', 12, 9);

INSERT INTO visita (data_inizio, data_fine, citta_id, utente_id) VALUES ('2012-12-31', '2013-02-28', 6, 10);

INSERT INTO commento (testo,data,citta_id,utente_id) VALUES ("Ammetto di avere un debole per Barcelona e forse di non essere perfettamente obbiettiva...ma questa città è meravigliosa!

Amate l'architettura? O l'arte? O la movida? La spiaggia? L'enogastronomia? Barcelona ha tutto questo e oltre...","2023-06-01 14:03:33",6,3);

INSERT INTO commento(testo,data,citta_id,utente_id) VALUES ("Amo Londra con tutto me stesso. Amo perdermi per le sue strade, il fatto che sia storica ma super moderna, la gente che si trova per le strade, ogni monumento che trasuda splendore.","2021-09-15 16:07:02",1,7);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Roma è una città ricca di storia e bellezza. Amo passeggiare per le strade di Trastevere e visitare il Colosseo. La cucina romana è deliziosa e non posso resistere a un piatto di pasta alla carbonara autentica.', '2016-09-12 19:26:18', 2, 6);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Firenze è una città che incanta con la sua arte e la sua architettura mozzafiato. Ammirare il Duomo di Santa Maria del Fiore e passeggiare lungo il Ponte Vecchio sono esperienze indimenticabili.', '2009-03-21 23:11:24', 12, 10);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Londra è una città incredibile, con una vasta storia e una cultura vibrante. Ho adorato visitare il British Museum e passeggiare lungo il Tamigi.', '2022-05-20 19:24:11', 1, 3);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Melbourne è fantastica, le sue strade colorate e gli artisti di strada mi hanno affascinato. Voglio proprio tornarci!', '2019-02-26 09:10:00', 4, 2);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Istanbul è una città magica che unisce Oriente e Occidente. La Moschea Blu e il Gran Bazar sono assolutamente da visitare.', '2017-02-10 16:15:26', 5, 4);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Tokyo è una metropoli affascinante e piena di vita. Ho adorato esplorare i quartieri di Shibuya e Akihabara.', '2020-12-09 15:23:12', 8, 4);

INSERT INTO commento (testo, data, citta_id, utente_id) VALUES ('Rio de Janeiro è una città mozzafiato! Le spiagge di Copacabana e Ipanema sono semplicemente incredibili. Ho apprezzato anche il calore e la passione contagiosa dei locali. Un luogo da non perdere!', '2019-10-23 17:12:08', 10, 3);

INSERT INTO info_generali (citta_id,sicurezza,abitanti,costo_connessione,costo_vita,stipendio,family_score,velocita_connessione) VALUES (1,4,9000000,15,6593,43646,4,16);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (2, 5, 2870000, 25, 5477, 25000, 3, 14);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (3, 2,289000, 30, 3589, 27685, 4, 42);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (4, 5, 5159200, 45, 7458, 54231, 5, 20);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (5, 3, 15200000, 13, 6842, 49785, 3, 15);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (6, 4, 5600000, 16, 6123, 44567, 4, 18);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (7, 4, 2141000, 20, 5432, 35000, 3, 13);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (8, 3, 9000000, 17, 6125, 43200, 4, 16);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (9, 2, 2074000, 29, 5790, 37600, 3, 12);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (10, 5, 5322000, 15, 6950, 51000, 5, 19);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (11, 3, 3659000, 22, 4923, 31500, 4, 15);

INSERT INTO info_generali (citta_id, sicurezza, abitanti, costo_connessione, costo_vita, stipendio, family_score, velocita_connessione) VALUES (12, 4, 383100, 24, 4387, 33600, 4, 22);

```
INSERT INTO info_meteo (citta_id,data,temperatura,umidita,nuvolosita,qualita_aria)
VALUES (1,CURDATE(),19,5,60,36);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (2, CURDATE(), 22, 7, 40, 42);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (3, CURDATE(), 16, 3, 20, 52);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (4, CURDATE(), 25, 8, 30, 48);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (5, CURDATE(), 30, 10, 25, 40);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (6, CURDATE(), 26, 6, 35, 45);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (7, CURDATE(), 24, 5, 55, 38);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (8, CURDATE(), 20, 4, 45, 42);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (9, CURDATE(), 28, 8, 25, 48);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (10, CURDATE(), 32, 7, 30, 35);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (11, CURDATE(), 18, 6, 50, 48);

INSERT INTO info_meteo (citta_id, data, temperatura, umidita, nuvolosita,
qualita_aria) VALUES (12, CURDATE(), 26, 3, 15, 52);


INSERT INTO lingua (nome) VALUES ('inglese');

INSERT INTO lingua (nome) VALUES ('italiano');

INSERT INTO lingua (nome) VALUES ('portoghese');

INSERT INTO lingua (nome) VALUES ('turco');

INSERT INTO lingua (nome) VALUES ('giapponese');

INSERT INTO lingua (nome) VALUES ('arabo');

INSERT INTO lingua (nome) VALUES ('catalano');

INSERT INTO lingua (nome) VALUES ('francese');

INSERT INTO lingua (nome) VALUES ('spagnolo');


INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (1, 1);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (2, 2);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (3, 3);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (1, 4);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (4, 5);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (7, 6);

INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (9, 6);
```

```
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (8, 7);
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (5, 8);
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (6, 9);
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (3, 10);
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (1, 11);
INSERT INTO lingua_parlata (lingua_id, citta_id) VALUES (2, 12);

INSERT INTO like_cit (citta_id,utente_id) VALUES (1,1);
INSERT INTO like_cit (citta_id,utente_id) VALUES (1,4);
INSERT INTO like_cit (citta_id,utente_id) VALUES (3,2);
INSERT INTO like_cit (citta_id,utente_id) VALUES (3,6);
INSERT INTO like_cit (citta_id,utente_id) VALUES (5,9);
INSERT INTO like_cit (citta_id, utente_id) VALUES (7, 3);
INSERT INTO like_cit (citta_id, utente_id) VALUES (8, 5);
INSERT INTO like_cit (citta_id, utente_id) VALUES (1, 2);
INSERT INTO like_cit (citta_id, utente_id) VALUES (3, 4);
INSERT INTO like_cit (citta_id, utente_id) VALUES (6, 1);
INSERT INTO like_cit (citta_id, utente_id) VALUES(4,1);

INSERT INTO like_comm (commento_id,utente_id) VALUES (1,2);
INSERT INTO like_comm(commento_id,utente_id) VALUES (1,4);
INSERT INTO like_comm (commento_id, utente_id) VALUES (1, 1);
INSERT INTO like_comm (commento_id, utente_id) VALUES (2, 3);
INSERT INTO like_comm (commento_id, utente_id) VALUES (3, 2);
INSERT INTO like_comm (commento_id, utente_id) VALUES (1, 3);
INSERT INTO like_comm (commento_id, utente_id) VALUES (2, 4);
INSERT INTO like_comm (commento_id, utente_id) VALUES (3, 5);
INSERT INTO like_comm (commento_id, utente_id) VALUES (4, 6);
INSERT INTO like_comm (commento_id, utente_id) VALUES (5, 7);
INSERT INTO like_comm (commento_id, utente_id) VALUES (6, 8);
INSERT INTO like_comm (commento_id, utente_id) VALUES (1, 9);
INSERT INTO like_comm (commento_id, utente_id) VALUES (2, 10);
INSERT INTO like_comm (commento_id, utente_id) VALUES (3, 1);
INSERT INTO like_comm (commento_id, utente_id) VALUES (9, 10);
INSERT INTO like_comm (commento_id, utente_id) VALUES (7, 4);
INSERT INTO like_comm (commento_id, utente_id) VALUES (2, 5);
INSERT INTO like_comm (commento_id, utente_id) VALUES (2, 7);
INSERT INTO like_comm (commento_id, utente_id) VALUES (5, 6);
INSERT INTO like_comm (commento_id, utente_id) VALUES (8, 1);
```

```
INSERT INTO like_comm (commento_id, utente_id) VALUES (3, 6);
```

5.3. Indici

```
CREATE INDEX stato_index ON citta(`stato`) ;
CREATE INDEX data_commento_index ON commento(`data`) ;
CREATE INDEX temperatura_index ON info_meteo(`temperatura`) ;
CREATE INDEX stipendio_medio_index ON info_generali(`stipendio`) ;
CREATE INDEX velocita_connessione_index ON info_generali(`velocita_connessione`) ;
```

5.4. View

```
CREATE VIEW info_meteo_citta AS
SELECT c.id AS id_citta,c.nome,c.stato,c.coordinate, im.id AS
id_info_meteo,im.data,im.temperatura,im.umidita,im.nuvolosita,im.qualita_aria
FROM citta AS c
JOIN info_meteo AS im ON im.citta_id = c.id;

CREATE VIEW info_generali_citta AS
SELECT c.id AS id_citta,c.nome,c.stato,c.coordinate,ig.id AS id_info_generali,
ig.sicurezza, ig.abitanti, ig. costo_connessione, ig.costo_vita, ig.stipendio, ig.
family_score, ig.velocita_connessione
FROM citta AS c
JOIN info_generali AS ig ON ig.citta_id = c.id;

CREATE VIEW numero_like_citta AS
SELECT c.nome,COUNT(lc.citta_id) AS like_città
FROM citta AS C
LEFT JOIN like_cit AS lc ON lc.citta_id = c.id
GROUP BY c.id,c.nome;

CREATE VIEW commenti_citta AS
SELECT cit.id AS id_citta, cit.nome, cit.stato, cit.coordinate, com.id AS id_commento,
com.testo, com.data, com.utente_id
FROM commento com
JOIN citta cit ON cit.id=com.citta_id;

CREATE VIEW numero_like_commento AS
```

```
SELECT com.id AS id_commento, COUNT(lc.commento_id) AS numero_like_commento
FROM commento AS com
JOIN like_comm AS lc ON com.id=lc.commento_id
GROUP BY com.id;
```

5.5. Query

QUERY RIGGI:

1. Trova l'id di tutti i commenti lasciati da "PicassoPablo" dopo il 2021 e il nome delle citta a cui fanno riferimento

```
SELECT commento.id as id_commento,citta.nome
FROM utente JOIN commento ON utente.id=commento.utente_id JOIN citta ON commento.citta_id=citta.id
WHERE utente.username="PicassoPablo" AND commento.data>="2022-01-01 00:00:00";
```

id_commento	nome
1	Barcelona
5	London

Tradotta in algebra relazionale:

$\Pi_{commento.id,citta.nome}($

$(\sigma_{utente.username="PicassoPablo"}(\rho_{utente.id,utente.email,utente.username,utente.password \leftarrow id,email,username,password}(utente)))$

$\bowtie_{utente.id=commento.utente\_id}$

$(\sigma_{com.data>="2022-01-01\ 00:00:00"}$

$(\rho_{commento.id,commento.testo,commento.data,commento.citta\_id,commento.utente\_id \leftarrow id,testo,data,citta\_id,utente\_id}(commento)))$

$\bowtie_{commento.citta\_id=citta.id}$

$(\rho_{citta.id,citta.nome,citta.stato,citta.coordinate \leftarrow id,nome,stato,coordinate}(citta))$

)

2. Trova gli username degli utenti che hanno effettuato visite in data 2023-06-01 e mostra oltre agli username anche il nome delle città visitate con i relativi dettagli della visita ordinando il risultato in base alla data di inizio della visita.

SELECT utente.username as utente, citta.nome as citta, visita.data_inizio , visita.data_fine
FROM visita JOIN utente ON visita.utente_id=utente.id JOIN citta ON visita.citta_id =citta.id
WHERE visita.data_inizio<="2023-06-01" and visita.data_fine>="2023-06-01"
ORDER BY visita.data_inizio;

utente	citta	data_inizio	data_fine
Regina	Barcelona	2023-01-15	2023-06-30
Jason_Statham	Melbourne	2023-05-01	2023-08-10

Tradotta in algebra relazionale:

$\Pi_{utente.username,citta.nome,visita.data\_inizio,visita.data\_fine}$ (

($\sigma_{visita.data\_inizio<="2023-06-01" \wedge visita.data\_fine>="2023-06-01"}$

($\rho_{visita.id,visita.data\_inizio,visita.data\_fine,visita.citta\_id,visita.utente\_id \leftarrow id,data\_inizio,data\_fine,citta\_id,utente\_id}$(visita))

)

$\bowtie_{visita.utente\_id=utente.id}$

($\rho_{utente.id,utente.email,utente.username,utente.password \leftarrow id,email,username,password}$(utente))

$\bowtie_{visita.citta\_id=citta.id}$

($\rho_{citta.id,citta.nome,citta.stato,citta.coordinate \leftarrow id,nome,stato,coordinate}$(citta))

)

3.Trova il numero totale di "mi piace" ricevuti da ogni città, inclusi i casi in cui non ci sono "mi piace", e ordina il risultato per numero di "mi piace" in ordine decrescente mostrando il nome di ogni città e il relativo numero di like ottenuti.

SELECT c.nome,COUNT(l.citta_id) as numeroLike
FROM citta c LEFT JOIN like_cit l ON c.id=l.citta_id
GROUP BY c.id
ORDER BY numeroLike DESC;

nome	numeroLike
London	3
Madeira	3
Melbourne	1
Istanbul	1
Barcelona	1
Paris	1
Tokyo	1
Roma	0
Cairo	0
Rio de Janeiro	0
Sydney	0
Firenze	0

4. Trova il nome delle città con stipendio>35000$, velocità connessione>15Mbps , costo connessione<25$, costo vita<6500$ e con almeno un like.

```
SELECT c.nome as Citta
FROM citta c JOIN info_generali i ON c.id=i.citta_id JOIN like_cit l ON c.id=l.citta_id
WHERE i.stipendio>35000 AND i.velocita_connessione>15 AND i.costo_connessione<25 AND i.costo_vita<6500
GROUP BY c.id,c.nome
HAVING COUNT(l.citta_id)>0;
```

Citta
Tokyo
Barcelona

5. Trova il nome delle città con temperatura maggiore di 20 gradi, family score pari a 5, in cui si parli l'inglese e con almeno un like.

```
SELECT  c.nome as citta
FROM citta c JOIN info_meteo im ON c.id=im.citta_id JOIN info_generali ig ON c.id=ig.citta_id JOIN like_cit l ON c.id=l.citta_id JOIN lingua_parlata lp ON c.id=lp.citta_id JOIN lingua ling ON lp.lingua_id=ling.id
```

```
WHERE im.temperatura>20 AND ig.family_score=5 AND ling.nome='inglese'
GROUP BY c.id,c.nome
HAVING COUNT(l.citta_id)>=1;
```

citta
Melbourne

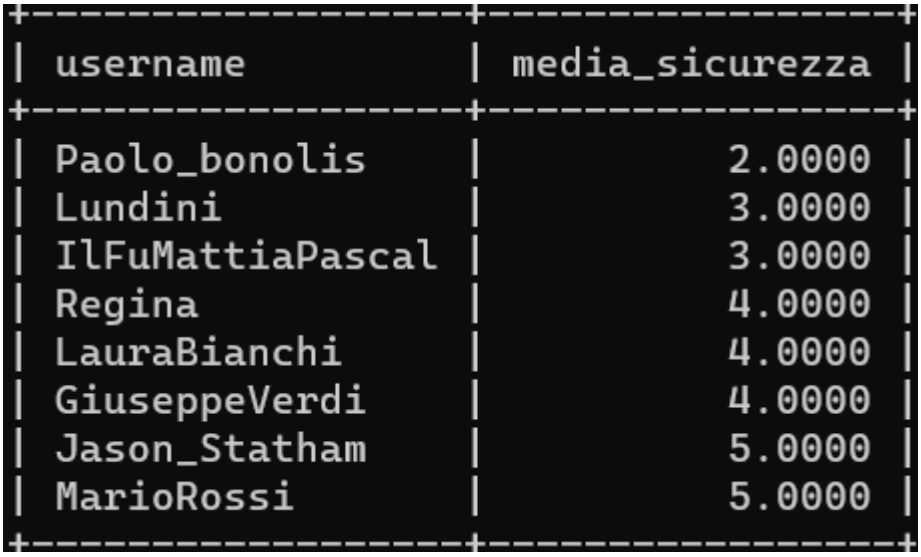
6. Trova l'username di tutti gli utenti che hanno lasciato almeno un like e almeno un commento alle città, con relativo numero di like e commenti lasciati. Ordina gli utenti in ordine alfabetico.

```
SELECT u.username as Utente,COUNT(l.utente_id) as NrLike,COUNT(c.utente_id) as NrCommenti
FROM utente u JOIN like_cit l ON u.id=l.utente_id JOIN commento c ON u.id=c.utente_id
GROUP BY u.id,u.username
ORDER BY u.username ASC;
```

Utente	NrLike	NrCommenti
Frassica	1	1
IlFuMattiaPascal	4	4
Paolo_bonolis	2	2
PicassoPablo	3	3

7. Per ogni utente che ha visitato almeno una citta trova il suo username e la media della sicurezza delle città che ha visitato. Ordina il risultato dal livello medio di sicurezza minore al maggiore.

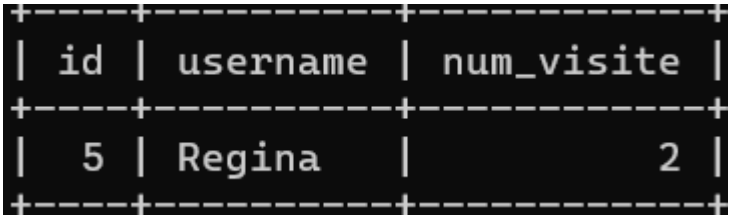
```
SELECT u.username,avg(i.sicurezza) as media_sicurezza
FROM citta c JOIN info_generali i ON c.id=i.citta_id JOIN visita v ON c.id=v.citta_id JOIN utente u ON u.id=v.utente_id
GROUP BY u.id,u.username
ORDER BY media_sicurezza ASC;
```



username	media_sicurezza
Paolo_bonolis	2.0000
Lundini	3.0000
IlFuMattiaPascal	3.0000
Regina	4.0000
LauraBianchi	4.0000
GiuseppeVerdi	4.0000
Jason_Statham	5.0000
MarioRossi	5.0000

8. Trova il numero di visite effettuate, l'id e l'username dell'utente che ha effettuato il maggior numero di visite

```
SELECT u.id, u.username, COUNT(v.utente_id) AS num_visite
FROM utente u JOIN visita v ON u.id = v.utente_id
GROUP BY u.id,u.username
ORDER BY num_visite DESC
LIMIT 1;
```



id	username	num_visite
5	Regina	2

9. Trova l'username degli utenti che hanno visitato almeno 2 città diverse e il numero di città visitate da ognuno. Il risultato deve avere gli username in ordine alfabetico.

```
SELECT u.username, COUNT(DISTINCT v.citta_id) as NrCittaVisitate
FROM utente u JOIN visita v on u.id=v.utente_id
GROUP BY u.username
HAVING COUNT(DISTINCT v.citta_id)>1
ORDER BY u.username;
```

username	NrCittaVisitate
Lundini	2
Regina	2

10. Trova l'id dell'utente che ha scritto il commento relativo a Roma che ha ricevuto più like (con relativo numero di like ottenuti).

```
SELECT com.utente_id AS IdAutore, COUNT(l.utente_id) AS num_like
FROM citta cit JOIN commento com ON cit.id=com.citta_id JOIN like_comm l ON com.id=l.commento_id JOIN utente u ON com.utente_id=u.id
WHERE cit.nome="Roma"
GROUP BY com.utente_id
ORDER BY num_like DESC
LIMIT 1;
```

IdAutore	num_like
6	4

QUERY PARIS:

1. Selezionare città che hanno una qualità dell'aria maggiore di 40

```
SELECT c.nome,im.qualita_aria
FROM citta AS C
JOIN info_meteo as im ON im.citta_id = c.id
WHERE im.qualita_aria >= 40;
```

nome	qualita_aria
Roma	42
Madeira	52
Melbourne	48
Istanbul	40
Barcelona	45
Tokyo	42
Cairo	48
Sydney	48
Firenze	52

$\Pi_{nome,qualita\_aria}($

$(\textbf{citta}) \bowtie_{id=citta\_id}$

$(\sigma_{qualita\_aria>=40}($

$\rho_{info\_meteo\_id \leftarrow id}($ ***info_meteo***$)))$

2. Selezionare i commenti delle città che hanno sicurezza >=4 mostrando nome della città, testo del commento e username dell'utente che ha scritto il commento

```
SELECT c.nome, com.testo, u.username
FROM citta AS c
JOIN commento AS com ON com.citta_id = c.id
JOIN utente AS u ON u.id = com.utente_id
JOIN info_generali AS ig ON ig.citta_id = c.id
WHERE ig.sicurezza >= 4;
```

nome	testo	username
Barcelona	Ammetto di avere un debole per Barcelona e forse di non essere perfettamente obbi...	PicassoPablo
London	Amo Londra con tutto me stesso. Amo perdermi per le sue strade, il fatto che sia sto...	Lundini
Roma	Roma è una città ricca di storia e bellezza. Amo passeggiare per le strade di Trastev...	Frassica
Firenze	Firenze è una città che incanta con la sua arte e la sua architettura mozzafiato. Amm...	GiuseppeVerdi
London	Londra è una città incredibile, con una vasta storia e una cultura vibrante. Ho adorato...	PicassoPablo
Melbourne	Melbourne è fantastica, le sue strade colorate e gli artisti di strada mi hanno affasci...	Paolo_bonolis
Rio de Janeiro	Rio de Janeiro è una città mozzafiato! Le spiagge di Copacabana e Ipanema sono se...	PicassoPablo

$\Pi_{nome,testo,username}($

$\rho_{commento\_citta\_id \leftarrow citta\_id}(\textbf{commento})$

$\bowtie_{commento\_citta\_id=id}(\textbf{citta})$

$\bowtie_{id=citta\_id}((\sigma_{sicurezza>=4}$

$\rho_{info\_generali\_id \leftarrow id}(\textbf{\textit{info\_generali}})))$

3. Selezionare nome degli utenti e nome delle città di tutte le visite ordinate per data di inizio

```
SELECT  u.username, cit.nome, v.data_inizio, v.data_fine
FROM utente AS u
JOIN visita AS v ON v.utente_id = u.id
JOIN citta as cit ON cit.id = v.citta_id
ORDER BY v.data_inizio ASC;
```

username	nome	data_inizio	data_fine
Regina	London	1930-03-12	2000-01-26
MarioRossi	Roma	2000-07-05	2000-08-20
GiuseppeVerdi	Barcelona	2012-12-31	2013-02-28
Lundini	Madeira	2017-06-16	2017-09-30
Lundini	Barcelona	2018-06-16	2018-07-10
IlFuMattiaPascal	Istanbul	2018-09-01	2018-09-15
Paolo_bonolis	Madeira	2022-09-10	2022-11-20
Regina	Barcelona	2023-01-15	2023-06-30
Jason_Statham	Melbourne	2023-05-01	2023-08-10
LauraBianchi	Firenze	2023-08-01	2023-08-15

Πusername,nome,data_inizio,da(

ρid_utente$\leftarrow$id(**utente**)

$\bowtie$id_utente =utente_id((**visita**)

$\bowtie$citta_id=id(***citta***)))

4. Seleziona il numero di visitatori unici per ogni città (conta una sola volta quelli che hanno visitato una città più volte), togliendo le città senza visitatori e ordinandole in base al numero di visitatori in ordine decrescente:

```
SELECT citta.nome, COUNT(DISTINCT visita.utente_id) AS visitatori_unici
FROM citta
JOIN visita ON visita.citta_id = citta.id
GROUP BY citta.id,citta.nome HAVING visitatori_unici > 0
ORDER BY visitatori_unici DESC;
```

nome	visitatori_unici
Barcelona	3
Madeira	2
Roma	1
Melbourne	1
London	1
Istanbul	1
Firenze	1

5. Selezionare il numero di like di ogni specifica città e restituisce anche le città che non hanno like

```
SELECT c.nome,COUNT(lc.citta_id) AS like_città
FROM citta AS C
LEFT JOIN like_cit AS lc ON lc.citta_id = c.id
GROUP BY c.id,c.nome
ORDER BY (like_città) DESC;
```

nome	like_città
London	3
Madeira	3
Melbourne	1
Barcelona	1
Tokyo	1
Istanbul	1
Paris	1
Roma	0
Rio de Janeiro	0
Firenze	0
Cairo	0
Sydney	0

6.Seleziona il numero di commenti per ogni utente, ordinati in modo decrescente per il numero di commenti:

```
SELECT u.username, COUNT(com.utente_id) AS num_commenti_utente
FROM utente as u
JOIN commento AS com ON com.utente_id = u.id
GROUP BY u.id,u.username
ORDER BY (num_commenti_utente) DESC;
```

username	num_commenti_utente
PicassoPablo	3
IlFuMattiaPascal	2
Lundini	1
GiuseppeVerdi	1
Paolo_bonolis	1
Frassica	1

7. Seleziona le città con il maggior numero di visite da parte degli utenti in ordine decrescente

```
SELECT c.nome, COUNT(v.citta_id) AS visita_citta
FROM citta as c
JOIN visita AS v ON v.citta_id = c.id
GROUP BY c.id,c.nome
ORDER BY visita_citta DESC;
```

nome	visita_citta
Barcelona	3
Madeira	2
Roma	1
Melbourne	1
London	1
Istanbul	1
Firenze	1

8. Seleziona la città con l'umidità massima registrata

```
SELECT c.nome, im.umidita
FROM citta AS c
JOIN info_meteo AS im ON im.citta_id = c.id
ORDER BY (im.umidita) DESC
LIMIT 1;
```

nome	umidita
Istanbul	10

9. Selezionare tutti i commenti e nome ed email dell'utente che l'ha scritto in ordine cronologico (dal più vecchio)

```
SELECT u.username, u.email, com.testo, com.data
FROM utente AS u
JOIN commento AS com ON com.utente_id = u.id
ORDER BY com.data ASC;
```

username	email	testo	data
GiuseppeVerdi	giuseppe.verdi@gmail.com	Firenze è una città che incanta con la sua arte e la sua architettura mozzafiato. Am...	2009-03-21 23:11:24
Frassica	nino.frassica@gmail.com	Roma è una città ricca di storia e bellezza. Amo passeggiare per le strade di Trastev...	2016-09-12 19:26:18
IlFuMattiaPascal	mattiapascal@gmail.com	Istanbul è una città magica che unisce Oriente e Occidente. La Moschea Blu e il Gra...	2017-02-10 16:15:26
Paolo_bonolis	paolobonolis@gmail.com	Melbourne è fantastica, le sue strade colorate e gli artisti di strada mi hanno affasci...	2019-02-26 09:10:00
PicassoPablo	pablopicasso@gmail.com	Rio de Janeiro è una città mozzafiato! Le spiagge di Copacabana e Ipanema sono se...	2019-10-23 17:12:08
IlFuMattiaPascal	mattiapascal@gmail.com	Tokyo è una metropoli affascinante e piena di vita. Ho adorato esplorare i quartieri d...	2020-12-09 15:23:12
Lundini	valerio.lundini@hotmail.it	Amo Londra con tutto me stesso. Amo perdermi per le sue strade, il fatto che sia sto...	2021-09-15 16:07:02
PicassoPablo	pablopicasso@gmail.com	Londra è una città incredibile, con una vasta storia e una cultura vibrante. Ho adorat...	2022-05-20 19:24:11
PicassoPablo	pablopicasso@gmail.com	Ammetto di avere un debole per Barcelona e forse di non essere perfettamente obbi...	2023-06-01 14:03:33

10. Seleziona le città con il numero di utenti che hanno lasciato un like a un commento riguardo quella città:

```
SELECT c.nome, COUNT(lc.commento_id) AS
num_utenti_che_hanno_lasciato_un_like_a_un_commento
FROM citta AS c
JOIN commento AS com ON com.citta_id = c.id
JOIN like_comm AS lc ON lc.commento_id = com.id
GROUP BY c.id,c.nome
ORDER BY (num_utenti_che_hanno_lasciato_un_like_a_un_commento) DESC;
```

nome	num_utenti_che_hanno_lasciato_un_like_a_un_commento
London	7
Barcelona	5
Roma	4
Istanbul	1
Tokyo	1
Firenze	1
Melbourne	1
Rio de Janeiro	1

11.Selezionare quante visite hanno fatto gli utenti e mettili in ordine decrescente:

```
SELECT u.username, COUNT(v.utente_id) AS numero_visite
```

```
FROM utente AS u
JOIN visita AS v ON v.utente_id = u.id
GROUP BY u.id,u.username
ORDER BY numero_visite DESC;
```

username	numero_visite
Lundini	2
Regina	2
GiuseppeVerdi	1
IlFuMattiaPascal	1
MarioRossi	1
Jason_Statham	1
LauraBianchi	1
Paolo_bonolis	1

12. Seleziona gli utenti che hanno visitato più città diverse e mettili in ordine decrescente:

```
SELECT u.username, COUNT(DISTINCT(v.citta_id)) AS citta_visitate
FROM utente AS u
JOIN visita AS v ON v.utente_id = u.id
GROUP BY u.id,u.username
ORDER BY citta_visitate DESC;
```

username	citta_visitate
Lundini	2
Regina	2
GiuseppeVerdi	1
IlFuMattiaPascal	1
MarioRossi	1
Jason_Statham	1
LauraBianchi	1
Paolo_bonolis	1

12. Ordina le città per numero di lingue parlate in modo decrescente

```
SELECT c.nome, COUNT(li.lingua_id) AS numero_lingue_parlate
FROM citta AS c
JOIN lingua_parlata AS li ON li.citta_id = c.id
GROUP BY c.id,c.nome
ORDER BY numero_lingue_parlate DESC
```

nome	numero_lingue_parlate
Barcelona	2
London	1
Sydney	1
Firenze	1
Rio de Janeiro	1
Tokyo	1
Melbourne	1
Roma	1
Madeira	1
Istanbul	1
Cairo	1
Paris	1