Basi di Dati (16AFQPL, 16AFQPI)

Anno Accademico 2025-2026

Politecnico di Torino

Utilizzo di Large Language Models per Text2SQL

Obiettivo

La finalità di questa esercitazione è quella di, dato uno schema logico-relazionale, generare coppie di domanda e risposta relative al task Text2SQL (ovvero formulare, a partire da un testo in linguaggio naturale, la corrispondente query SQL).
L'esercitazione prevede di generare coppie di domanda e risposta in lingua italiana relative a pattern di difficoltà differenti, testando le abilità di tre diversi Large Language Models (LLMs) e confrontando le soluzioni generate con diversi prompt con la soluzione attesa.

Descrizione della base di dati

La base di dati da utilizzare è denominata *Biblioteca* e raccoglie le informazioni relative alla gestione di una biblioteca ed è progettata per gestire e tenere traccia di libri, autori, lettori, prestiti, dipendenti e prenotazioni delle sale.

Schema logico relazionale:
LIBRO(id_libro, *id_autore*, titolo, anno_pubblicazione, genere, numero_copie)
AUTORE(id_autore, nome, cognome, nazionalità, data_nascita, data_morte*)
LETTORE(id_lettore, nome, cognome, email, telefono, indirizzo, data_iscrizione)
PRESTITO_LIBRO(*id_libro*, *id_lettore*, data_prestito, data_restituzione*, note*)
SALA(id_sala, *id_dipendente_responsabile*, nome, capacità, piano, tipologia)
DIPENDENTE(id_dipendente, nome, cognome, ruolo, data_assunzione, email, telefono)
PRENOTAZIONE_SALA(*id_lettore*, data, ora_inizio, ora_fine, *id_sala*, note*)

Le chiavi primarie sono sottolineate e le chiavi esterne sono in *corsivo*.
L'asterisco indica i campi opzionali.

La tabella **Libro** raccoglie le informazioni su ogni libro disponibile nella biblioteca. Ogni libro è identificato da un codice univoco (`id_libro`) ed è caratterizzato dall'autore (`id_autore`), titolo (`titolo`) e anno di pubblicazione (`anno_pubblicazione`), genere letterario (`genere`) e numero di copie disponibili (`numero_copie`). Si suppone che un autore possa scrivere più libri ma che un libro sia scritto da un unico autore.

La tabella **Autore** memorizza le informazioni degli autori. Ogni autore è identificato da un codice univoco (`id_autore`) ed è caratterizzato dal nome (`nome`), cognome (`cognome`), nazionalità (`nazionalità`), data di nascita (`data_nascita`) ed eventuale data di morte (`data_morte`).

La tabella **Lettore** contiene le informazioni degli utenti registrati alla biblioteca. Ogni lettore è identificato da un codice univoco (`id_lettore`) ed è caratterizzato dal nome (`nome`), cognome (`cognome`), indirizzo email (`email`), numero di telefono (`telefono`) e indirizzo di residenza (`indirizzo`). Inoltre, viene registrata la data di iscrizione alla biblioteca (`data_iscrizione`).

La tabella **Prestito_Libro** registra i prestiti dei libri effettuati dai lettori. Ogni prestito è identificato dalla combinazione dell'identificativo del libro prestato (`id_libro`), del lettore che lo ha richiesto (`id_lettore`) e dalla data di inizio prestito (`data_prestito`). Si memorizza inoltre la data di restituzione (`data_restituzione`) ed eventuali note o informazioni aggiuntive (`note`).

La tabella **Sala** raccoglie le informazioni sulle sale presenti nella biblioteca. Ogni sala è identificata da un codice univoco (`id_sala`) e ha un dipendente responsabile assegnato (`id_dipendente_responsabile`). Le sale sono caratterizzate da un nome (`nome`), una capacità massima di persone (`capacità`), il piano in cui si trovano (`piano`) e la tipologia di utilizzo (`tipologia`).

La tabella **Dipendente** memorizza le informazioni sui lavoratori della biblioteca. Ogni dipendente è identificato da un codice univoco (`id_dipendente`) ed è caratterizzato dal nome (`nome`), cognome (`cognome`), ruolo svolto nella biblioteca (`ruolo`), data di assunzione (`data_assunzione`), indirizzo email (`email`) e numero di telefono (`telefono`).

Infine, la tabella **Prenotazione_Sala** registra le prenotazioni delle sale effettuate dai lettori. Ogni prenotazione è identificata dalla combinazione dell'identificativo del lettore che l'ha prenotata (`id_lettore`) e dalla data (`data`) e ora di inizio (`ora_inizio`) della prenotazione. Si registra inoltre l'ora di fine della prenotazione (`ora_fine`), l'identificativo della sala prenotata (`id_sala`) ed eventuali note o informazioni aggiuntive sulla prenotazione (`note`).

Svolgimento

Durante il laboratorio, completando eventualmente l'esercitazione autonomamente in un secondo momento, a ciascuno studente è richiesto di:

- Creare **una domanda** (in lingua italiana) **per ogni tipologia di pattern** indicata nella pagina successiva. Essa rappresenta una query espressa in linguaggio naturale risolvibile utilizzando il pattern in analisi.
- Scrivere la **soluzione attesa** (in SQL) alla query proposta.
- Formulare i **prompt testuali** da inviare all'LLM.

  Occorre partire dalla struttura del prompt fornito nell'ultima pagina e completarlo per ogni tipologia di pattern richiesto. I prompt devono essere scritti in **lingua italiana** e devono includere lo schema logico delle tabelle della base dati.

  Nota: Per alcuni pattern (3, 6, 7), per rispondere alla stessa domanda occorre preparare versioni diverse del prompt in cui chiedere esplicitamente all'LLM di risolvere la query utilizzando una determinata strategia (es. tramite Join, IN, EXISTS, INTERSECT).
- Interrogare **i tre Large Language Models** utilizzando i prompt generati:
  - **GPT-5.3:** https://chatgpt.com/
  - **Gemini 3:** https://gemini.google.com/
  - **Qwen3.6-Plus:** https://chat.qwen.ai/

  Nota: Prima di interrogare Gemini 3 e Qwen3.6-Plus, selezionare dall'interfaccia web (dal menu a tendina vicino alla barra della chat) la versione "Veloce" del modello. Per le query più complesse, provare eventualmente a valutare l'output del modello selezionando le versioni "Ragionamento"/"Pro" (per Gemini 3) e "Pensiero" (per Qwen3.6-Plus).
- Salvare l'output (in SQL) fornito dai vari LLM per poterlo successivamente analizzare.
- **Analizzare** le risposte fornite dai modelli, confrontandole con la soluzione attesa per valutarne la correttezza, eventuali errori o modalità di risoluzione diverse.

Per poter utilizzare i vari modelli, è consigliato registrarsi tramite la propria e-mail personale (es. utilizzando l'indirizzo email PoliTo) o effettuare l'accesso tramite account terzi (es. Google).

Al termine dell'esercitazione, si suggerisce di formulare un commento generale sulle principali risultanze sperimentali evidenziate tra cui, ad esempio:

- Punti forti e punti deboli di ciascun LLM
- Confronti tra LLM
- Analisi e confronti tra tipologie diverse di prompt

Pattern

Le query proposte prevedono livelli di difficoltà diversa, secondo i seguenti pattern:

0. **Proiezione**: la (vostra) soluzione attesa alla domanda proposta deve contenere la clausola WHERE, più eventualmente ORDER BY.

1. **Join**: la (vostra) soluzione attesa alla domanda proposta deve prevedere l'operazione di JOIN, più eventualmente altre condizioni di selezione.

2. **Raggruppamento con condizione e calcolo di funzione aggregata**: la (vostra) soluzione attesa alla domanda proposta deve contenere le clausole GROUP BY e HAVING e deve prevedere il calcolo di una funzione aggregata.

3. **Soluzioni multiple Join/IN/EXISTS/INTERSECT**: la domanda proposta deve poter essere risolta utilizzando strategie diverse su cui interrogare separatamente l'LLM:
    - Join
    - IN
    - EXISTS
    - INTERSECT

4. **Divisione**: la (vostra) soluzione attesa alla domanda proposta deve prevedere un'operazione di divisione.

5. **Soluzioni multiple NOT IN/NOT EXISTS/EXCEPT**: la domanda proposta deve poter essere risolta utilizzando strategie diverse su cui interrogare separatamente l'LLM:
    - NOT IN
    - NOT EXISTS
    - EXCEPT

6. **Soluzioni multiple TABLE FUNCTION/CTE/Correlazione**: la domanda proposta deve poter essere risolta utilizzando strategie diverse su cui interrogare separatamente l'LLM:
    - TABLE FUNCTION
    - CTE
    - Correlazione

Esempio di prompt

Pattern 0 – Proiezione

Sei un assistente AI per la risoluzione di query in linguaggio SQL partendo da una domanda testuale.
Ti verranno forniti in input lo schema logico delle tabelle da utilizzare e la domanda testuale.
Tu dovrai restituire come output solamente il codice SQL.

Tabelle:

LIBRO(id_libro, id_autore, titolo, anno_pubblicazione, genere, numero_copie) con chiave primaria id_libro
AUTORE(id_autore, nome, cognome, nazionalità, data_nascita, data_morte*) con chiave primaria id_autore
LETTORE(id_lettore, nome, cognome, email, telefono, indirizzo, data_iscrizione) con chiave primaria id_lettore
PRESTITO_LIBRO(id_libro, id_lettore, data_prestito, data_restituzione*, note*) con chiave primaria (id_libro, id_lettore, data_prestito)
SALA(id_sala, id_dipendente_responsabile, nome, capacità, piano, tipologia) con chiave primaria id_sala
DIPENDENTE(id_dipendente, nome, cognome, ruolo, data_assunzione, email, telefono) con chiave primaria id_dipendente
PRENOTAZIONE_SALA(id_lettore, data, ora_inizio, ora_fine, id_sala, note*) con chiave primaria (id_lettore, data, ora_inizio)

L'asterisco indica i campi opzionali.

Domanda:

Mostra il titolo e l'anno di pubblicazione dei libri del genere "fantasy" con almeno 3 copie disponibili ordinati a partire dal più recente.

Codice SQL: