

Domande esame (possibili)

Si descriva brevemente cosa siano indipendenza logica e fisica dai dati e quale/i modello di basi di dati consente/ono di realizzarle

L'architettura a livelli di un DBMS garantisce l'indipendenza dei dati in due livelli:

- **Indipendenza fisica:** consente di interagire con il DBMS in modo indipendente dalla struttura fisica dei dati, senza influire sulle descrizioni e quindi sui programmi che usano i dati
- **Indipendenza logica:** consente di interagire con il livello esterno della base di dati in modo indipendente dal livello logico, per esempio come aggiungere un nuovo schema esterno senza modificare lo schema logico e perciò la sottostante organizzazione fisica dei dati

Il **modello relazionale** risponde perfettamente al requisito dell'indipendenza dei dati perché introduce una netta distinzione tra il livello fisico e quello logico. A differenza dei modelli precedenti (reticolare e gerarchico), che includevano riferimenti esplicativi a puntatori fisici, il modello relazionale è basato su valori e non richiede la conoscenza delle strutture fisiche per accedere ai **dati**

"Descrivere brevemente cosa sono le ACID properties, a cosa servono e da dove deriva l'espressione 'ACID'" oppure "Illustrare brevemente cosa siano le proprietà ACID (spiegando anche l'acronimo) ed a quale entità/oggetto sono riferite"

Un DBMS prevede che le interazioni con la base di dati avvengano per mezzo di **transazioni**.

Una **transazione** è una sequenza di azioni di lettura e scrittura del DB e di elaborazioni di dati in memoria temporanea, che il DBMS esegue garantendo le seguenti proprietà (ACID properties):

- **Atomicity:** La transazione è eseguita nella sua interezza oppure non è eseguita affatto (le transazioni che terminano prematuramente sono abortite)
- **Consistency preservation:** una esecuzione corretta della transazione porta il DB da uno stato consistente all'altro (i vincoli di integrità devono essere rispettati).
- **Isolation:** L'esecuzione di una transazione deve essere indipendente da quella di altre transazioni concorrenti. Il risultato deve essere analogo a quello che si otterrebbe eseguendo le transazioni una alla volta (serialmente).
- **Durability :** le modifiche su DB di una transazione terminata normalmente sono permanenti, cioè non sono alterabili da malfunzionamenti successivi alla terminazione

Si definisca brevemente cosa siano i trigger, il paradigma su cui sono basati, vantaggi e svantaggi per essi

I trigger, detti anche regole attive, sono dei costrutti per rendere la base di dati in grado di reagire a eventi definiti dall'amministratore tramite l'esecuzione di opportune azioni.

Seguono il paradigma Evento-Condizione-Azione (ECA), ossia:

- **Evento:** Tipicamente una modifica dello stato del database consiste in una operazione di INSERT, DELETE, UPDATE. Se l'evento accade, il trigger si attiva.
- **Condizione:** Predicato booleano espresso in SQL che identifica se l'azione del trigger deve essere eseguita. Quando il trigger si attiva, viene valutata la condizione, quindi il trigger viene considerato.
- **Azione:** Consiste in una sequenza di update SQL o una procedura. Quando la condizione è verificata, allora l'azione viene eseguita, quindi il trigger viene eseguito.

I trigger sono uno strumento molto potente che permette di gestire vincoli di integrità, calcolare dati derivati, gestire eccezioni e codificare regole aziendali. Un ulteriore vantaggio derivante dall'utilizzo dei trigger consiste nel riuscire a codificare la logica del sistema in maniera centralizzata e condivisa da tutte le applicazioni, con conseguenti vantaggi in fase di lettura e manutenzione del codice, infatti in caso di modifiche al comportamento del sistema è sufficiente intervenire nell'ambito della definizione dei trigger e non in più parti del codice.

Lo svantaggio è che i trigger sono standardizzati solo per SQL-3, per cui potrebbero presentarsi casi (se pur sempre più rari) di non portabilità del codice.

Illustrare brevemente cosa sia il conflitto di impedenza e quali soluzioni esistano per gestirlo

Un importante problema che caratterizza l'integrazione tra SQL e i normali linguaggi di programmazione è il cosiddetto **conflitto di impedenza**. I linguaggi di programmazione accedono agli elementi di una tabella scandendone le righe una a una (tuple-oriented). Al contrario SQL è un linguaggio di tipo set-oriented, che opera su intere tabelle e restituisce come risultato di un'interrogazione un'intera tabella. Le soluzioni a questo problema si ottengono con l'utilizzo dei **cursori** e l'utilizzo di linguaggi con costruttori di tipo in grado di gestire una struttura del tipo "insieme di righe" (Call Level Interface).

Si definiscano brevemente le nozioni di sistema organizzativo, sistema informativo e sistema informatico e si delinei la differenza esistente tra di essi

Un **sistema organizzativo** è un insieme di risorse e regole che consentono il funzionamento di una qualunque struttura sociale per il raggiungimento dei suoi obiettivi

Ogni sistema organizzativo è dotato di **sistema informativo**, ossia l'insieme delle risorse e delle procedure che un'organizzazione utilizza per gestire le informazioni necessarie al

perseguimento dei propri scopi.

Un sistema informativo è molte volte indipendente dalle automatizzazioni che conosciamo.

Un **sistema informatico** non è altro che la parte automatizzata di un sistema informativo, quindi un sistema software orientato alla gestione dei dati, dove l'aspetto prevalente è rappresentato dai dati stessi (memorizzati, ricercati, modificati) che costituiscono il patrimonio informativo di un'**organizzazione**

Si descriva brevemente i/il criteri/o di ottimizzazione delle query implementato dal gestore di ottimizzazione delle query di un DBMS

[da finire]