# Test di fine settimana – Week 4

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Michela |
|  |  | Cognome | Murtas |
|  |  | Data | 16/07/2021 |

*Leggete attentamente ogni domanda e argomentare quanto più possibile* ***fornendo anche degli esempi****.  
ATTENZIONE: Le domande a risposta multipla possono contenere più risposte corrette.*

1. ***Spiegare brevemente la differenza tra i comandi SELECT, INSERT, UPDATE E DELETE e fare un esempio per ognuno***

* Con il Comando **SELECT** è possibile visualizzare e recuperare tutti i dati inseriti.
* Con **INSERT**è possibile invece inserire i dati in una tabella
* **UPDATE**aggiorna i dati che sono stati precedente inseriti
* Mentre con **DELETE** i dati vengono cancellati

ESEMPI. Ho una tabella che contiene i dati riguardanti dei Gelati venduti in una gelateria

**SELECT** \* **FROM** gelato

*(in questo modo sto visualizzando TUTTI i dati senza alcuna distinzione*

*Se avessi voluto visualizzare solo nome e prezzo al posto di \* avrei potuto mettere* ***nomeGelato*** *e* ***prezzoGelato*** *separati da una virgola)*

**INSERT INTO** gelato (nomeGelato, marcaGelato, prezzoGelato)

**VALUES** (‘Cornetto’,4, 1)

*(in questo caso sto inserendo nelle colonne nomeGelato, marcaGelato e prezzoGelato - contenute nella tabella gelato - dei nuovi dati*

*Con il 4 su marcaGelato sto dando una FOREIGN KEY che si riferisce all’ID della marca)*

**UPDATE** gelato **SET** prezzoGelato = 2

**WHERE** prezzoGelato > 5

*(aggiornamento dei prezzi dei gelati. Aggiorna TUTTI i prezzi superiori a 5€ (ladri) abbassandoli a 2€.*

*Senza il* ***WHERE*** *avrei modificato tutti i prezzi, sempre utilizzando questo esempio)*

**DELETE FROM** gelato

*(in questo modo sto cancellando TUTTI i file contenuti su gelato, perché sto omettendo la condizione di* ***WHERE****)*

***Disegnare un esempio di tabelle con relazione 1:N e un esempio di relazione N:N e spiegare quali sono le differenze***

Abbiamo sempre la nostra Gelateria con le tabelle Gelato, Marca, Negozio.

**UN Gelato** appartiene a **UNA SOLA Marca**, ma **UNA Marca** ha ovviamente **PIU’ Gelati**. Questo è un caso di relazione **1:N.**

Durante la distribuzione in Negozio è possibile che **PIU’ Negozi** vendano **PIU’ Marche** e viceversa. Questa dovrebbe essere una relazione **N:N**, che si semplifica con l’utilizzo di una **Bridge Table** che contiene gli **ID** delle Marche e dei Negozi.

***Spiegare la differenza tra una PRIMARY KEY e una FOREIGN KEY***

La **PRIMARY KEY** è il dato **UNIVOCO** di una tabella. Non può avere doppioni e serve a distinguere un determinato dagli altri.  
 La **FOREIGN KEY** funziona allo stesso modo ma è una chiave ‘estranea’ appartenente ad un’altra tabella.

Tornando alla Gelateria, nella tabella Gelato possiamo avere come **PK** ‘CodiceGelato’ e come **FK** ‘marcaGelato’ che proviene dalla tabella Marca.

***Quando si utilizza l’istruzione “GROUP BY”? Fare un esempio pratico comprensivo di query SQL***

Con **GROUP BY** vengono raggruppati i dati per una determinata condizione. Se ad esempio volessi raggruppare i gelati per la Marca scriverei:

**SELECT** nomeGelato **FROM** gelato

**GROUP BY** marcaGelato

***Cos’è un Constraint? Fornire 2 esempi di uso in SQL***

I **CONSTRAINT** sono letteralmente dei **VINCOLI** che servono per determinare le regole all’interno di una tabella.

Ad esempio quando stiamo creando una tabella si può definire la **NULLABILITA’** di dati - sempre tra i gelati il CodiceGelato NON può essere NULL in quanto PK, ma se andiamo in un’ipotetica colonna Allergeni (?!) potrebbe non contenere alcun dato.

Un altro vincolo può essere il CHECK dove viene settato un una sorta di controllo. Ad esempio scrivendo: prezzoGelato int **CHECK** (prezzoGelato >5)

Non potrei inserire prezzi maggiori di 5€.

***Cos’è una Stored Procedure? Quali sono i casi in cui conviene ricorrere ad essa?***

Sono codici che possono essere riutilizzati più volte, come ad esempio la creazione di una tabella o l’inserimento dei dati in essa contenuta.

***Esercitazione pratica***

*Si vuole realizzare un sistema informativo per automatizzare la gestione di un negozio di dischi.*

*Le entità coinvolte (con i relativi attributi) sono:*

*Album:*

* *Titolo*
* *Anno di uscita*
* *Casa discografica*
* *Genere*
* *Supporto di distribuzione*

*Brano:*

* *Ttitolo*
* *Durata (espressa in secondi)*

*Band:*

* *Nome*
* *NumeroComponenti*

*È possibile che uno stesso brano faccia parte di più di un album (ad es. le raccolte contengono brani appartenenti, in genere, ad album già pubblicati).*

*Individuare la soluzione più adatta a livello di tabelle e creare tutte le relazioni necessarie.*

*Implementare i seguenti vincoli:*

* *Gli id devono essere autoincrementali*
* *Un album deve essere considerato unico sulla base del titolo, anno di uscita, casa editrice e genere e supporto (se uno stesso album viene memorizzato su, ad esempio, due supporti differenti, i dati relativi a quell’album devono essere registrati separatamente).*
* *Il genere può essere di queste tipologie: Classico, Jazz, Pop, Rock, Metal*
* *Il supporto di distribuzione deve essere scelto tra: CD, Vinile, Streaming*

*Una volta realizzato il modello entità-relazionale realizzare le seguenti query SQL:*

1. *Scrivere una query che restituisca i titoli degli album di Franco Battiato;*
2. *Selezionare tutti gli album editi dalla casa editrice nell’anno specificato;*
3. *Scrivere una query che restituisca tutti i titoli delle canzoni dei U2 appartenenti ad album pubblicati prima del 1990;*
4. *Individuare tutti gli album in cui è contenuta la canzone “Imagine”;*
5. *Restituire il numero totale di canzoni eseguite dai Pooh;*
6. *Contare per ogni album, la somma dei minuti dei brani contenuti*

*Una delle query (a scelta) deve essere realizzata come Stored Procedure con parametri.*

*Creare una view che mostri i dati completi dell’album, della band e dei brani contenuti in esso.*

*Scrivere una funzione utente che calcoli per ogni genere musicale quanti album sono inseriti in catalogo.*

*Caricare la prova pratica e teorica su Github. Per la parte pratica, caricare gli script SQL necessari a ricreare il modello, le query, la view e la funzione.*