# Test di valutazione – Modulo 3

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Carlotta |
|  |  | Cognome | Colla |
|  |  | Data | 12/02/2021 |

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. *Quali di queste affermazioni sul DBMS è vera?*

è un motore che gestisce la creazione e l’interrogazione del database

deve essere installato nella stessa macchina in cui è presente anche “Sql Server Management Studio” (SSMS)

è ottimizzato per gestire l’interrogazione del database in maniera efficiente

è un software client

1. *Se voglio memorizzare un valore numerico con virgola quali tipi posso utilizzare?*

int

bit

decimal

string

È possibile utilizzare anche una stringa ma poi non può essere utilizzato per effettuare dei calcoli.

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

Il comando SELECT serve per selezionare quale colonne visualizzare nel risultato delle query, es. SELECT movies.titolo, movies.durata FROM movies, mostrerà solo le colonne titolo e durata della tabella movies, mentre con SELECT \* FROM movies, vengono mostrate tutte le colonne della tabella.

Con INSERT è possibile inserire i dati nelle tabelle, si possono aggiungere tutti i valori di un record attraverso INSERT INTO movies VALUES (valori), altrimenti è possibile specificare quali valori inserire specifincando i nomi delle colonne, ad esempio INSERT INTO movie (titolo, durata) VALUES (valori), così verranno inseriti solo il titolo e la durata (l’id se è autoincrementale non viene inserito con l’insert ma viene aggiunto in automatico).

Con UPDATE è possibile modificare il valore di un campo di un singolo record o tutti i valori di una colonna, ad esempio: UPDATE movies SET titolo = ‘Narnia’ WHERE id = 3, in questo modo viene modificato solo il titolo del film con id 3, mentre senza la calusola where (UPDATE movies SET titolo = ‘Narnia’) vengono modificati tutti i titoli dei film presenti nella tabella.

DELETE serve per eliminare un singolo record o l’intera tabella: DELETE FROM movies WHERE id=3, elimina solo il record con id=3 all’interno della tabella movies, senza la clausola where (DELETE FROM movies) vengono eliminati tutti i record della tabella.

1. *Quali sono i vantaggi di utilizzare una “stored procedure” anziché le query SQL?*

Le stored procedure, rispetto alle query, sono statement sql che vengono salvati e possono essere richiamati in qualsiasi parte del codice. Servono per calcoli complessi, possono contenere delle query e richiamare delle funzioni.

Possono avere dei parametri di input e dei valori di output, per questo possono essere utili come supporto alle query.

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

Nelle relazioni 1:N in una tabella (1) è presente una chiave secondaria collegata all’id della seconda tabella (N)

Es. 1:N Uno studente può essere iscritto a un solo corso di laurea

A un corso di laurea possono essere iscritti piu studenti

|  |  |
| --- | --- |
| *ID* | *Corso di laurea* |
| *2* | *Informatica* |

|  |  |  |
| --- | --- | --- |
| *ID* | *Matricola* | *ID\_Corso di laurea* |
| *01* | *277600* | *2* |

Nelle relazioni N:N deve essere presente una tabella ponte che contiene le due chiavi secondarie collegate agli id delle due tabelle.

Es. N:N Uno studente può iscriversi più volte allo stesso esame

|  |  |
| --- | --- |
| *ID* | *Descrizione esame* |
| *4* | *Algoritmi 1* |

A un esame possono iscriversi più studenti

|  |  |
| --- | --- |
| *ID* | *Matricola* |
| *01* | *277600* |

|  |  |
| --- | --- |
| *ID\_Studente* | *ID\_Esame* |
| *01* | *4* |

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

La primary key (o chiave primaria) è un campo della tabella univoco e non nullo, questo campo serve per identificare univocamente un record, solitamente è un numero intero autoincrementale per semplicità.

Mentre una foreign key è una chiave secondaria, sempre univoca e non nulla perchè si collega alla chiave primaria di un’altra tabella. Prima di scrivere un record contente la chiave secondaria è importante scrivere il record con la chiave primaria altrimenti non viene rispettata l’integrità referenziale.

1. *Date le seguenti tabelle che identificano l’associazione tra alunni di una scuola e le relative classi scrivere la “SELECT con JOIN” per ottenere l’elenco degli alunni con le informazioni della classe a loro assegnata*

*Alunno*

* *ID (int primary key)*
* *Nome (varchar)*
* *Cognome*
* *ClasseID (int foreign key)*

*Classe*

* *ID (int primary key)*
* *Anno (int)*
* *Sezione (char)*

SELECT \* FROM Alunno

LEFT JOIN Classe

ON Alunno.ClasseID = Classe.ID

Con select \* mi ritornano tutti i campi delle due tabelle.

Utilizzo un left join perchè voglio che mi ritorni l’elenco completo di tutti gli alunni, anche se non hanno una classe assegnata.

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

Il group by viene utilizzato quando nel select si utilizzano delle funzioni che hanno bisogno di un raggruppamento, come ad esempio la somma dei campi con SUM(), il campo minimo con MIN() o la conta dei record con COUNT().

Es. Sommare i posti occupati di un cinema:

tabella prenotazioni

|  |  |  |
| --- | --- | --- |
| ID | Film | Posti\_occupati |
| 1 | Narnia | 3 |
| 2 | La mummia | 10 |
| 3 | Narnia | 5 |

SELECT prenotazione.Film, SUM(prenotazione.Posti\_occpati)

*FROM prenotazioni*

*GROUP BY prenotazione.Film*

In questo modo si sommano i posti occupati dei film che hanno lo stesso nome grazie al group by.

1. Scrivere la query SQL per generare una “stored procedure” che permette di aggiornare un singolo record nella tabella “users” che ha i seguenti campi:

* Id (int primary key)
* name (varchar)
* surname (varchar)
* birthdate (date)

CREATE PROCEDURE stpAggiornaUnCampo @ID INT, @nomeCampo NVARCHAR(255), @valoreCampo NVARCHAR(255)

AS

BEGIN

IF (@nomCampo = ‘name’)

UPTADET user SET name= @valoreCampo WHERE Id = @ID

IF (@nomCampo = ‘surname’)

UPTADET user SET surname= @valoreCampo WHERE Id = @ID

IF (@nomCampo = ‘birthdate’)

UPTADET user SET birthdate= CAST(@valoreCampo DATE) WHERE Id = @ID

RETURN 0;

END

Questa procedura prende come parametri di inputi l’id per poter modificare il record giusto, il nome del campo che voglio modificare e il valore con cui voglio modificarlo.

Se il campo è da modificare è birthdate è necessario effettuare un cast per trasformare il valore inserito (NVARCHAR) in una data (DATE).

Il campo ID non può essere modificato per rispettare l’integrità referenziale.

1. *Esercitazione pratica*

*creare il database "POLIZIA" per gestire l'associazione tra agenti di polizia e le aree metropolitane che devono pattugliare (può essere che un agente venga assegnato a più aree)*

*entità coinvolte in questo esercizio:*

*AGENTE DI POLIZIA:*

*- nome (stringa obbligatoria di massimo 30 caratteri)*

*- cognome (stringa obbligatoria di massimo 50 caratteri)*

*- codice fiscale (stringa di 16 caratteri obbligatoria)*

*- data di nascita (data obbligatoria)*

*- anni di servizio (valore numerico valorizzato con gli effettivi anni di servizio)*

*AREA METROPOLITANA*

*- codice area (stringa alfabetica di 5 caratteri che identifica l'area (non è l'id))*

*- alto rischio (valore che può assumere "0" o "1" a seconda se l'area è considerata ad alto rischio)*

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

*IMPLEMENTARE I SEGUENTI VINCOLI:*

*-gli id devono essere autoincrementali*

*-l'agente di polizia deve essere maggiorenne*

*-il codice fiscale non può essere duplicato*

*IMPLEMENTARE LE SEGUENTI STORED PROCEDURE:*

*- inserimento di un nuovo agente(prendere in input i dati necessari)*

*- eliminazione di un agente esistente*

*- visualizzare l'elenco degli agenti che lavorano in "aree ad alto rischio" e hanno meno di 3 anni di servizio*

*- visualizzare il numero di agenti assegnati per ogni area geografica (numero agenti e codice area)*