בס"ד

**מבחן קבלה – כלל ביטוח**

שם: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

תאריך: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

משך המבחן: שלוש שעות

המבחן מנוסח בלשון זכר, אך פונה לנשים ולגברים כאחד.

יש להעביר את המבחן בסיום הזמן שהוקצה גם אם לא סיימתם את כל המשימות.

בכל שאלה אפשר להתקשר לנייד : 052-6582615(תהילה) או 054-5706821 (דוד)

בהצלחה!

# **חלק א' – שאלות תאורטיות**

1. נתונות הטבלאות הבאות

טבלת לקוחות - CRM.Customer

|  |  |  |
| --- | --- | --- |
| מספר רץ | Int | ID |
| מספר ת.ז | String | IdentityNo |
| שם מלא | String | FullName |
| תאריך יצירה | Datetime | CreateDate |

טבלת כיסויים - CRM.PolicyDetails

|  |  |  |
| --- | --- | --- |
| מספר רץ | Int | ID |
| לקוח | Int – מפתח לטבלת הלקוחות | CustomerId |
| סטטוס 4= פעיל | Int | StateCode |
| סכום ביטוח | decimal | InsuranceAmt |

(לא נדרש להקים טבלאות או נתונים אלא לרשום את השאילתה בלבד, בהינתן שהטבלאות מלאות במיליוני רשומות )

* יש להציג במסך את 30 הלקוחות שיש להם מעל 4 כיסויים ושהסך הכולל של סכום הביטוח שלהם הוא מעל 2000 ₪
* יש להציג בעמוד את השדות הבאים: ID, IdentityNo , FullName ,מספר הכיסויים הפעילים ללקוח והסך הכולל של סכום הביטוח.

SELECT TOP 30 crm.Customer.id,

crm.Customer.IdentityNo,

crm.Customer.FullName,

COUNT(crm.PolicyDetails.InsuranceAmt) AS CntInsuranceAmt

FROM crm.Customer INNER JOIN

crm.PolicyDetails ON crm.Customer.id = crm.PolicyDetails.CustomerId

WHERE (crm.PolicyDetails.StateCode >= 4)

GROUP BY crm.Customer.id, crm.Customer.IdentityNo, crm.Customer.FullName

HAVING (SUM(crm.PolicyDetails.InsuranceAmt) >= 2000)

4. בהינתן קטע קוד הבא, מה המשמעות של השימוש בבלוק using ? איזה אובייקטים ראוי להגדיר תחת הבלוק, ומה קורה כאשר הבלוק מסתיים?

using (Font font1 = new Font("Arial", 10.0f))

{

byte charset = font1.GdiCharSet;

}

This Code is equivalent (would be translated to IL )to :

Font font1 = new Font("Arial", 10.0f);

try

{

byte charset = font1.GdiCharSet;

}

finally

{

font1.Dispose();

}

So dispose accrues after byte charset = font1.GdiCharSet; operator in all the case yes or no have been exception;

5. שימוש בטבלאות זמניות ב SQL .

* אילו סוגים אתה מכיר?
* מתי כדאי להשתמש בכל סוג?
* באילו מצבים אפליקטיביים תעדיף להשתמש בכל סוג, אם בכלל?
* A temp table is a real database table in a permanent database.  All temp tables reside in the tempdb database, which is a system database.  A temp table is temporary in that it is generally no longer available when the database connection for creating a temp table no longer exists.
* Subqueries are select statements nested inside of other SQL statements or clauses of SQL statements.  There are many different variations for subqueries, and these variations can affect the application for a subquery.
* A derived table is a special kind of subquery inside of the from clause of a select statement.  This type of subquery is code that returns a result set that acts as a table in many respects.
* A table variable has some properties like local variables and other properties like tables.  You define a table variable with a declare statement.  Its scope is the batch, stored procedure, or function in which the table declaration resides.
* A common table expression (CTE) is a result set returned by a select statement.  It is temporary in the sense that its scope is limited to the trailing select, insert, delete, or update statement.  CTEs are often used to simplify complex queries.  Also, CTEs are sometimes used for recursive queries – namely, for queries that call themselves.
* Another strategy for storing temporary data is in a permanent table.  With this approach, you populate a table in a user database, then operate on it as required, and finally drop the permanent table when the temporary data it stores is no longer needed, such as at the end of an application.

6 . מה ההבדל בין Observable לבין Promise

|  |  |
| --- | --- |
| **Observables** | **Promises** |
| Emit multiple values over a period of time. | Emit a single value at a time. |
| Are lazy: they’re not executed until we subscribe to them using the subscribe() method. | Are not lazy: execute immediately after creation. |
| Have subscriptions that are cancellable using the unsubscribe() method, which stops the listener from receiving further values. | Are not cancellable. |
| Provide the map for forEach, filter, reduce, retry, and retryWhen operators. | Don’t provide any operations. |
| Deliver errors to the subscribers. | Push errors to the child promises. |

7. Dependency Injection

* תאר בקצרה מהו Dependency Injection ומה חדש ב .CORE
* תאר איך ניתן להזריק Dependency ל Controller

.NET Core provides four kinds of dependency injection, based in your lifetimes:

* Transient: services that will be created each time they are requested
* Scoped: services that will be created once per client request (connection)
* Singleton: services that will be created only at the first time they are requested
* System: services that will be created once at application creation

Project below contains some DI servers :

**חלק ב' – מעשי**

כתוב מסך ANGULAR המכיל את האובייקטים הבאים:

1. DROPDOWN של עובדים המציג שם ותמונה

2. רשימת מחלקות בארגון לפי היררכיה

3. טבלת משימות

* יש לממש את הדפים ע"י פניה לSERVICE שמחזיר את האוביקטים בהתאם לבחירה
* יש להקים דף JSON המכיל את נתוני העובדים, מחלקות, משימות כמופיע להלן (אפשר להרחיב אם רוצים)
* [{“EmployeeName”:”name here”,”EmployeePicture”:”piture path here”}]
* [{“DepartmentID”:number,“DepartmentName”:”Name here”,”ParentID”:number}]
* [{“TaskNumber”:number,”TaskName”:”name”,”StatusID”:Number,” EmployeeName “:”Employee Name”,” DepartmentID “:number,”DueDate”:date}]

4. מימוש :

* טבלת משימות
  + צריכה להציג את המשימות שתאריך היעד שלהן עבר (DueDate) באדום.
  + יש להציג על המסך רק 10 רשומות
    - לאפשר הגדלת שינוי מספר הרשומות
    - לאפשר מעבר בין עמודים ע"י גלילה וירטואלית או Paging רגיל
  + יש להקפיא את שורת הכותרת כך שאם יש הרבה רשומות במסך וגוללים מטה תמיד יראו את הכותרת.
* יש להציג רק מחלקות ראשיות (ParentID=0) ואם יש להם בנים לאפשר לחיצה על שם המחלקה לראות את הבנים (מימוש TREE).
* בבחירה של עובד יש להציג את המשימות שלו בטבלה
* בבחירה של מחלקה יש להציג את המשימות של המחלקה בטבלה

