# Data Modeling Questions

* Design a database to model a cookbook with meal types, recipes, ingredients.
  1. You can use paper to document your data model.

**Recipe\_Steps\_Ingredients**

(PF) recipe\_id

(PF) step\_number

(PF) ingredient\_id

Measurement\_req.

**DB\_Cookbook**

**Meal\_Courses**

(PF) Meal\_id

(Brk,Lun,din,dess,snk)

Meal\_name

(PK) course\_number

Course\_name

**Recipes**

(PK) recipe\_id

recipe\_name

recipe\_description

**Ingredients**

(PK) indgredients\_id

(FK) ingredient\_type\_code

ingredient\_name

ingredient\_mesurments

**Course\_Recipe\_Choices**

(PF) menu\_id

(PF) course\_number

(PK) choice\_number

(FK) recipe\_id

**Recipes\_Steps**

(PF) recipe\_id

(PK) step\_name

recipe\_instructions

# SQL Questions

* Given the following SQL SELECT statement what are the highlighted text represent?

SELECT ***CUST***.CUSTOMER AS ***NAME***, ***CUST***.CUSTOMER\_ADDRESS

FROM CUSTOMER\_V ***CUST***

WHERE ***NAME*** = ‘Home Furnishings’;

I believe they represent the entity column aliases names.

Given the following Orders and Customers tables:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | | **CustomerID** | **OrderDate** | |
| 10308 | | 2 | 1996-09-18 | |
| 10309 | | 37 | 1996-09-19 | |
| 10310 | | 77 | 1996-09-20 | |
| **CustomerID** | **CustomerName** | | | **ContactName** | | **Country** |
| 1 | Alfreds Futterkiste | | | Maria Anders | | Germany |
| 2 | Ana Trujillo Emparedados y helados | | | Ana Trujillo | | Mexico |
| 3 | Antonio Moreno Taquería | | | Antonio Moreno | | Mexico |

What would the result set be given the following SQL SELECT statement?

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
INNER JOIN Customers  
ON Orders.CustomerID=Customers.CustomerID;

**OrderID CustomerName OrderDate CustomerID**  10308 Ana Trujillo Emparedados y helados 1996-09-18 2

What would the result set be given the following SQL SELECT statement?

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
LEFT JOIN Customers  
ON Orders.CustomerID=Customers.CustomerID;

Same Question and Answer as above.

# Entity Framework Questions

* What do the following Entity Framework Object Services provide for your application in regards to data from a database?
  1. Materialization – allows you to materialize data returned from the database as entity objects.
  2. Change Tracking – allows you to track changes that were made to entity objects.
  3. Object identities – determines the identity of an entity and performs identity resolution in the objects context.

Design a code first data model which has a Project class that can contain a bunch of tasks.

public class Blog

{

public int BlogId { get; set; }

public string Name { get; set; }

public virtual List<Post> Posts { get; set; }

}

public class Post

{

public int PostId { get; set; }

public string Title { get; set; }

public string Context { get; set; }

public int BlogId { get; set; }

public virtual Blog Blog { get; set; };

}

# Object Orientation Questions

* What are the basic concepts of OOP? Objects, Classes, Inheritance, Abstraction, Encapsulation, Polymorphism, Reusability, Overloading
* How do you program behavior into your C# class? You can program behaviors through Methods declared in a class or struct by specifying access levels such as public and private.
* Explain method overriding. Method overriding allows you to invoke functions that belong to different classes in the same hierarchy of inheritance using the base class reference.
* What is Inheritance?Inheritance is creating a new class from an existing class.
* What is abstract class? An Abstract class is a special type of class that cannot be instantiated and acts as base class for other classes.

# MVC Questions

What is an example URL that would call the following controller method, assuming the default routes have been configured?

/Catalog/Specifications

Part II: What HTTP Verb is used?

public class CatalogController : Controller

 {

     public ActionResult Specifications(int id)

     {

         var model = new SpecModel(id);

         return View();

     }

}

Examine the following View for an MVC Application:

@model IEnumerable<MVCGuidedLab.Models.Color>

@{

    ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

    @Html.ActionLink("Create New", "Create")

</p>

<table class="table">

    <tr>

        <th>

            @Html.DisplayNameFor(model => model.Name)

        </th>

        <th>

            @Html.DisplayNameFor(model => model.Value)

        </th>

        <th></th>

    </tr>

@foreach (var item in Model) {Go

    <tr>

        <td>

            @Html.DisplayFor(modelItem => item.Name)

        </td>

        <td>

            @Html.DisplayFor(modelItem => item.Value)

        </td>

        <td>

            @Html.ActionLink("Edit", "Edit", new { id=item.Id }) |

            @Html.ActionLink("Details", "Details", new { id=item.Id }) |

            @Html.ActionLink("Delete", "Delete", new { id=item.Id })

        </td>

    </tr>

}

</table>

1. What type is the Model?
2. Is the model a single object, or a sequence?
3. What properties are on the items in the model?

# Programming Exercises

To turn in this exam, you’ll create a fork of a repository, make modifications to the project in that repository, and submit a pull request with your changes.

We’ll walk you through the github workflow for those pieces. However, the code will be yours.

Go to Github.com and fork the repository <https://github.com/BillWagner/ExperienceITExam>

Then, clone your fork to your desktop. You’ll do that by clicking the “Clone in Desktop” button on the github page.

Now, you’re ready to do your work.

1. Open the program.cs file in the Loops project. The Main method has comments that describe what you should do. Make your changes.
2. Open the program.cs file in the Extension methods project. The Main method has comments that describe the code you should add. Make your changes.
3. Open the program.cs file in the Lazy Evaluation project. The Main method has comments that describe the code you should add. Make those changes as well.
4. Open the program.cs file in the Query Expresssions project. As before, the Main method has comments that describe the code you should or change.
5. Make a new ASP.NET MVC project and add it to the solution. Add a controller called DiceRoll controller. Modify the index method and the index view to display all the combinations of results from rolling 2 six-sided dice. (For example, { 1, 1}, {1, 2} etc.

Commit your changes, and then push them to github.

Now, you’re ready to submit a pull request. Navigate to your fork of the repository in github.com in a browser. Below the Code tab on the right side, you’ll see a link that says “Pull Request”. Click that. Once you’re on the Pull Request page, submit a new pull request. Add your name, and any comments you would like on your pull request, and submit the request.

We can now look at your changes, and see how you did.