**Interview Questions and Answers**

**Q. What is the Business Object Runtime in RAP ABAP?**

**A:** In the context of the ABAP RESTful Application Programming Model (RAP), the runtime of a RAP business object (BO) consists of two main parts: the interaction phase and the save sequence

**Interaction Phase:**

During this phase, the RAP BO handles requests from external clients (such as OData clients).

It processes read and write operations, including validations and authorization checks.

The interaction phase ensures that data is consistent and adheres to business rules.

**Save Sequence:**

After successful interaction, the save sequence comes into play.

It handles the persistence of data changes (e.g., inserts, updates, or deletions) to the underlying database.

The save sequence ensures that data modifications are correctly applied and committed.

**Q. What is EML in the RAP ABAP context?**

**A.**Entity Manipulation Language (EML) provides new syntaxes in ABAP to control the behavior of business objects created using the RAP model.

It allows developers to manipulate transactional data within the context of RAP BOs (Business Objects).

EML syntaxes cover actions like reading, modifying (create, update, delete), and committing data.

**Q. Your team is developing a managed RAP Fiori application for purchase orders. A key requirement is to ensure that purchase orders can only be created for future dates (today or later). Backdated purchase orders (creation date in the past) should be rejected. The Fiori application should display clear and informative error messages to the user if they attempt to create a backdated purchase order. How will you handle this scenario?**

**A.**we can use **Validation** to handle the mentioned scenario.

VALIDATION *validate\_purchase\_date* ON SAVE { CREATE }

Implement the method *validate\_purchase\_date* in the local handler class and write validation logic in it.

**Q. What is validation in RAP ABAP and how it works?**

**A.** In RAP (ABAP RESTful Application Programming Model), validation refers to the process of ensuring that data submitted to a business object (BO) adheres to specific rules and constraints. This helps maintain data consistency and integrity within your application.

Validations are typically triggered during specific events, such as CRUD Operations (Create, Read, Update, Delete) and Field Changes. Validations are always invoked during the SAVE sequence.

An invoked validation can reject inconsistent instance data from being saved by passing the keys of failed instances to the corresponding table in the FAILED structure.

Additionally, a validation can return messages to the consumer by passing them to the corresponding table in the REPORTED structure.

**Q. The Fiori list report displays a list of sales orders. The team wants to add an "Approve Order" on the Fiori screen. Clicking this button should trigger an action to update the sales order status from "Pending" to "Approved." RAP (RESTful Application Programming Model) is being used to build the backend services for this Fiori application. How can you implement the "Approve Order" button using RAP?**

**A**. we can use **Action** to handle the mentioned scenario.

**CDS behavior definition:**

ACTION *approve\_order*RESULT [1] $SELF

**Annotation:**

@UI: {lineitem: [ { type: #FOR\_ACTION

                                dataAction: 'approve\_order'

                                label: 'Approve Order'  }   ] }

**Behavior Projection:**

USE ACTION approve\_order

**Behavior Implementation:**

Implement the method *approve\_order* in the local handler class and write modify logic in it.

**Q. When to use Action in RAP ABAP?**

**A.** In the context of ABAP RESTful Application Programming Model (RAP), an action is a non-standard operation that allows you to change the data of a business object (BO) instance.

Actions provide flexibility beyond standard CRUD (Create, Read, Update, Delete) operations.

You can define custom business logic specific to your application requirements.

Examples include triggering workflows, calculating derived fields, status updates etc.