Guidewire important Questions with Answers part-3:

**1. What is delegate?**

Delegates are reusable components that provide common functionality to multiple entities without requiring inheritance. They allow different entities to share behavior without duplicating code. If you are creating delegate entity, we can use that delegate entity to mention in our entity –< implements Entity > implement delegate entity we use all the fields and behavior of that delegate entity in out normal entity.

**Features:**

* **Reusable** – Provides shared functionality across multiple entities.
* **Avoids Code Duplication** – Reduces the need for redundant code.
* **Flexible** – Can be attached to multiple entities as needed.
* **Not Inheritance-Based** – Unlike direct entity inheritance, entities "implement" delegates rather than extend them.

**2. What is supertype?**

A supertype entity is a data model entity which has one or more other entities that act as subtypes. In a supertype/subtype entity structure:

* The top-level entity is referred to as **the parent entity or the supertype entity.**
* Each lower-level entity is referred to as **a child entity or a subtype entity**.

**3. What is subtype?**

Subtype entities are child entities of a supertype entity or parent entity. All the fields in the parent entity are inherited by the child entity. A top-level supertype and all the child subtypes are stored in a single database table.

Subtype:

1. In Studio, navigate to **configuration** > **config** > **Extensions** > **Entity**.
2. Right-click **Entity**, and then click **New** > **Entity.**
3. In the **Entity** text box, type Inspector\_Ext.
4. In the **Entity Type** drop-down list, click **subtype**.
5. In the **Supertype** box, type Person.
6. Click **OK**.

**4. View Entity:**

A View Entity is a read-only virtual entity used in Guidewire to display data from one or more related entities, especially in List Views or search screens. It improves performance and simplifies UI design.

For example, to show ClaimNumber, LossDate, and InsuredName in a single table, I can create a View Entity that joins Claim and Policy data using path attributes.

**5. Non persistent Entity:**

A Non-Persistent Entity in Guidewire is a temporary, in-memory data holder that is not saved in the database. It’s used for passing data in UI forms, search criteria, or wizard steps.

For example, I created a ClaimSearchCriteria entity to hold Claim Number, Loss Date, and Insured Name on a search screen. When the user hits “Search,” the inputs are used to filter real Claim data, but the criteria themselves are never stored.

**4. What is bundle?**

* A Bundle is a collection of entities that are being created, modified, or deleted within a single transaction.
* Once all changes are made, the Bundle is committed, and the changes are saved to the database. If an error occurs, the entire Bundle can be rolled back to maintain data consistency.

**5. Types of bundles:**

A bundle is a unit of work that ensures data integrity and consistency when performing transactions. It groups multiple operations (such as database updates) into a single transaction, ensuring that all changes are committed together or rolled back if any issue occurs.

**1.runWithNewBundle():**

The runWithNewBundle() method is used to execute a block of code within a separate new transaction (bundle). This ensures that the operations within the block do not interfere with the current user session’s bundle.

gw.transaction.Transaction.runwithnewbundle(\bundle->{

bundle.add()

get.current()

})

**2.newBundle():**

The newBundle() method is used to create a completely new transaction (bundle) where database operations can be performed independently of the current bundle. This is useful when you want to perform operations without affecting the existing transaction in the user's session.

**3.readOnlyBundle():**

A ReadOnlyBundle is a special type of bundle that allows data retrieval but prevents modifications. It is used when you want to fetch data without the risk of accidentally changing it.

**4.copyBundle():**

The copyBundle() method is to creates a duplicate (copy) of the current transaction bundle. This allows you to work with the same set of entity data in a separate transaction while maintaining its state.

**Contact:**

Contact is the person, company, vendor or place related to the claim. They are stored in Contact Manager.

**Adjudication:**

Adjudication is the process of determining if the claim will be paid, and if so, how much you are going to pay. It includes everything from when the claim is first reported, to issuing the final payment and closing the claim.

**Inbound invoice:**

An **inbound invoice** is a bill that your company **receives** from **someone else** (usually a vendor, repair shop, law firm, or third party).

A body shop sends an invoice for repairing a claimant’s car.

**Outbound invoice:**

An **outbound invoice** is a bill that your company **sends** to **someone else** (usually a client, customer, or another business unit).

You (an insurer or TPA) send an invoice to a reinsurer or policyholder for reimbursement.