**Azure Deployment Workflow - Study Guide**

**1. ARM vs Bicep Export**

* **ARM (Azure Resource Manager)**:
  + JSON-based format for defining Azure resources.
  + Can be verbose and harder to manually edit.
  + Directly supported by Azure Portal and CLI.
* **Bicep**:
  + Higher-level DSL for ARM templates.
  + Simplifies syntax, supports modularity and reusability.
  + Easier to read, write, and maintain.
* **Export Process**:
  + From Azure Portal: Export ARM templates of existing resources.
  + From ARM to Bicep: Use bicep decompile to convert.
* **Best Practice**: Use Bicep as the source of truth; export ARM for compatibility.

**2. Parameter Files**

* **Purpose**: Store environment-specific variables outside of the main template.
* **Advantages**:
  + Separate configuration from logic.
  + Reuse templates across dev, test, and prod.
* **Structure**:
  + JSON for ARM.
  + .parameters.json for Bicep.
* **Best Practice**:
  + Keep secrets out of parameter files; store them in Key Vault.

**3. Linked IR Tokens**

* **Integration Runtime (IR)**: Enables data movement and transformation in Azure Data Factory or Synapse.
* **Linked IR Tokens**:
  + Used for secure cross-region or cross-subscription IR usage.
  + Token-based authentication mechanism.
* **Best Practice**:
  + Store tokens in Azure Key Vault.
  + Use Managed Identity wherever possible to reduce token management complexity.

**4. Build → Release Pipeline**

* **Build Stage**:
  + Validate templates (Bicep linting, ARM schema validation).
  + Package artefacts for deployment.
* **Release Stage**:
  + Deploy to a staging or test environment first.
  + Run smoke tests.
  + Include manual approval gates for production.
* **Tools**:
  + Azure DevOps Pipelines.
  + GitHub Actions.
* **Best Practice**: Fail fast in the build stage to prevent bad deployments.

**5. Smoke Test**

* **Purpose**: Quick functional check after deployment.
* **Scope**:
  + Confirm critical resources are online.
  + Validate a single representative workflow.
* **Automation**: Implement as a pipeline stage after deployment.
* **Best Practice**: Keep tests minimal and fast for early detection of issues.

**6. Back-out Plan**

* **Purpose**: Ensure quick recovery from a failed deployment.
* **Methods**:
  + Redeploy last known good template.
  + Use Azure deployment history for rollbacks.
* **Best Practice**:
  + Maintain versioned deployment artefacts.
  + Document rollback procedures.
  + Test rollback processes regularly.