

Salesforce Plat-Arch-202

**Salesforce Certified Platform Development Lifecycle and
Deployment Architect**

For More Information – Visit link below:

<https://www.examsempire.com/>

Product Version

- 1. Up to Date products, reliable and verified.**
- 2. Questions and Answers in PDF Format.**



<https://examsempire.com/>

Visit us at: <https://www.examsempire.com/plat-arch-202>

Latest Version: 24.0

Question: 1

What are two limitations an architect should consider when designing a strategy for managing technical reference data, with multiple related objects? Choose 2 answers

- A. HTTP response size
- B. Apex CPU limits
- C. Circular relationships
- D. Depth of nested relationships

Answer: C, D

Explanation:

The correct selection is C, D. Technical reference-data migrations are constrained by relationship shape. Circular references cannot be represented cleanly as one rooted hierarchy, and nested object trees have a finite supported depth, so complex reference structures often need staged loads and follow-up relationship updates. From a deployment perspective, the same release should be reproducible across environments, with dependencies, metadata/data differences, API behavior, and target-org constraints identified before production. Automation is valuable because it turns deployment knowledge into a controlled process rather than relying on memory or manual reconstruction. The rejected alternatives do not meet the deployment requirement because they introduce drift, ignore target-org compatibility, or substitute a convenience tool for a reproducible release artifact. For this scenario, the decisive point is therefore the platform behavior represented by the selected options (C. Circular relationships; D. Depth of nested relationships). Applying that rule consistently keeps the implementation aligned with Salesforce-supported lifecycle practices and makes the resulting change easier to review, validate, and operate across environments.

Study Guide Deploying — Metadata API and Tooling API; deployment validation; package/change-set deployment; data migration; dependency management.

Question: 2

Salesforce has three major releases a year.

Which type of change introduced by a release can cause automated browser tests to need updating?

- A. New Apex methods
- B. Metadata schema changes
- C. DOM changes

D. New standard fields

Answer: C

Explanation:

The correct selection is C. Automated browser tests depend on the rendered Document Object Model. Salesforce can change internal Lightning markup between seasonal releases even when the business behavior is unchanged, so brittle selectors and DOM assumptions can require test maintenance. From a testing perspective, the goal is deterministic evidence that the solution behaves correctly under the conditions that matter. Tests should isolate external dependencies, use controlled data, exercise positive and negative paths, and reproduce realistic volume or user behavior where required without making production the first place defects are discovered. The alternatives do not provide equivalent evidence: they either validate only a narrow implementation detail or depend on conditions outside the test's control. For this scenario, the decisive point is therefore the platform behavior represented by the selected option (C. DOM changes). Applying that rule consistently keeps the implementation aligned with Salesforce-supported lifecycle practices and makes the resulting change easier to review, validate, and operate across environments.

Study Guide Testing — Apex testing; test-data isolation; mocks; regression testing; performance/load testing; sandbox test strategy.

Question: 3

Universal Containers is adopting Scrum as an agile methodology and wants to choose a software tool to support the adoption.

What three key features of an agile development support tool should an architect look for?

Choose 3 answers

- A. Product backlog prioritization
- B. Sprint backlog management
- C. Work (for example, user stories or tasks) assignment
- D. Kanban board
- E. Email notifications when work is created or changed

Answer: A, B, C

Explanation:

The correct selection is A, B, C. Scrum tooling must support the ordered product backlog, sprint backlog, and assignment or ownership of actionable work. Those capabilities map directly to Scrum planning and execution, whereas email alerts or a generic Kanban view are supporting conveniences rather than the core Scrum controls being asked for. From a system-design perspective, the decision should minimize unnecessary coupling while preserving clear ownership, scalability, governance, and extensibility. Salesforce architecture is strongest when boundaries reflect real business and technical responsibilities instead of using organizational

convenience as the primary design driver. The other options introduce tighter coupling, duplicate governance, or solve a different architectural concern and therefore do not satisfy the stated boundary and ownership requirements. For this scenario, the decisive point is therefore the platform behavior represented by the selected options (A. Product backlog prioritization; B. Sprint backlog management; C. Work (for example, user stories or tasks) assignment). Applying that rule consistently keeps the implementation aligned with Salesforce-supported lifecycle practices and makes the resulting change easier to review, validate, and operate across environments.

Study Guide System Design — org strategy; modular architecture; integration topology; data ownership; enterprise governance; scalability.

Question: 4

The team at Universal Containers is building an application on Java that will interact with its Salesforce application. They want to use SOQL queries to retrieve and make changes to smaller pieces of Salesforce metadata through this application. Which API should the team leverage?

- A. Metadata API
- B. User Interface API
- C. Tooling API
- D. Any Salesforce API

Answer: C

Explanation:

The correct selection is C. Tooling API is intended for fine-grained programmatic access to development and metadata-related resources and exposes many of them as queryable objects. That makes it better suited than package-oriented Metadata API when a Java application needs targeted SOQL-driven metadata operations. From a deployment perspective, the same release should be reproducible across environments, with dependencies, metadata/data differences, API behavior, and target-org constraints identified before production. Automation is valuable because it turns deployment knowledge into a controlled process rather than relying on memory or manual reconstruction. The rejected alternatives do not meet the deployment requirement because they introduce drift, ignore target-org compatibility, or substitute a convenience tool for a reproducible release artifact. For this scenario, the decisive point is therefore the platform behavior represented by the selected option (C. Tooling API). Applying that rule consistently keeps the implementation aligned with Salesforce-supported lifecycle practices and makes the resulting change easier to review, validate, and operate across environments.

Study Guide Deploying — Metadata API and Tooling API; deployment validation; package/change-set deployment; data migration; dependency management.

Question: 5

Universal Containers' architect is documenting the application lifecycle management (ALM) process to communicate it to the development teams from different Implementation partners. Which three steps apply to any Salesforce development project? Choose 3 answers

- A. Build Release
- B. Test
- C. Develop
- D. Continuous Integration.
- E. Change Sets

Answer: A, B, C

Explanation:

The correct selection is A, B, C. Develop, Test, and Build Release are lifecycle activities that apply regardless of the specific Salesforce deployment technology. Continuous Integration is an engineering practice, while change sets are one deployment mechanism, so neither is a universal lifecycle stage for every project. From an ALM perspective, the controlling principles are a clear source of truth, repeatable change tracking, controlled promotion, testable release artifacts, and traceability from requirement through production. Mature teams separate the lifecycle from any single tool so the process remains consistent even when the deployment mechanism changes. The alternatives fail because they either create manual tracking or treat one delivery technology as the lifecycle, which weakens auditability and makes releases harder to reproduce. For this scenario, the decisive point is therefore the platform behavior represented by the selected options (A. Build Release; B. Test; C. Develop). Applying that rule consistently keeps the implementation aligned with Salesforce-supported lifecycle practices and makes the resulting change easier to review, validate, and operate across environments. Study Guide Application Lifecycle Management — ALM stages; source control; change tracking; package/org development models; CI/CD; release traceability.

Thank You for Trying Our Product

Special 16 USD Discount Coupon: **NSZUBG3X**

Email: support@examsempire.com

**Check our Customer Testimonials and ratings
available on every product page.**

Visit our website.

<https://examsempire.com/>