Exam4Future

Certified Data Architecture and Management Designer

Salesforce Certified-Data-Architecture-and-Management-Designer

Version Demo

Total Demo Questions: 24

Total Premium Questions: 243

Buy Premium PDF

https://exam4future.com

support@exam4future.com

Topic Break Down

Topic	No. of Questions
Topic 1, Data Modeling/Database Design	30
Topic 2, Master Data Management	30
Topic 3, Salesforce Data Management	62
Topic 4, Data Governance	41
Topic 5, Large Data Volume	74
Topic 6, Mix Questions	6
Total	243

QUESTION NO: 1

UC has millions of Cases and are running out of storage. Some user groups need to have access to historical cases for up to 7 years.

Which 2 solutions should a data architect recommend in order to minimize performance and storage issues?

Choose 2 answers:

A. Export data out of salesforce and store in Flat files on external system.

B. Create a custom object to store case history and run reports on it.

C. Leverage on premise data archival and build integration to view archived data.

D. Leverage big object to archive case data and lightning components to show archived data.

ANSWER: C D

Explanation:

“Leverage on premise data archival and build integration to view archived data.” is appropriate because it moves older, infrequently accessed Case records out of the core Salesforce transactional data store. This reduces Salesforce data-storage consumption and helps keep operational Case queries, reports, and automation focused on the active data set. An integration can provide authorized users with controlled, on-demand access to the retained historical records while the external archive applies the organization’s required retention, security, and retrieval policies.

“Leverage big object to archive case data and lightning components to show archived data.” is also appropriate for high-volume historical data that must remain available from Salesforce. Big Objects are designed to store and query massive volumes of records without consuming standard object data storage in the same way as regular objects. A carefully designed index supports predictable retrieval patterns, and a Lightning component can present relevant archived Case information to users in the Salesforce experience. Together, these approaches separate seven years of historical data from the active Case workload while preserving access for the user groups that need it. See Salesforce’s Big Objects overview and Big Objects documentation.

QUESTION NO: 2

UC migrating 100,000 Accounts from an enterprise resource planning (ERP) to salesforce and is concerned about ownership skew and performance.

Which 3 recommendations should a data architect provide to prevent ownership skew?

Choose 3 answers:

A. Assigned a default user as owner of accounts, and assign role in hierarchy.

B. Keep users out of public groups that can be used as the source for sharing rules.

C. Assign a default user as owner of account and do not assign any role to default user.

D. Assign “view all” permission on profile to give access to account.

E. Assign a default user as owner of accounts and assigned top most role in hierarchy.

ANSWER: B C E

Explanation:

Ownership skew occurs when one user owns a very large number of records, commonly more than 10,000, and Salesforce must evaluate access through that owner’s role hierarchy. For a large Account migration, assigning a default user as owner of account and do not assign any role to default user avoids role-hierarchy implicit sharing calculations for the high-volume owner. This is Salesforce’s preferred pattern when a single integration, migration, or queue-like owner must hold many records.

Assigning a default user as owner of accounts and assigned top most role in hierarchy is also an accepted hierarchy strategy when the owner must have a role. A role at the top of the hierarchy limits the number of users above the owner who receive implicit access, reducing the sharing impact associated with the high-volume ownership model.

Keeping users out of public groups that can be used as the source for sharing rules helps contain the sharing model around these heavily owned Accounts. Public-group-based sharing can expand access and increase sharing maintenance work; keeping those groups narrowly scoped reduces the performance pressure that can accompany ownership skew during bulk loads and ongoing updates. Salesforce recommends designing ownership and sharing structures to minimize skew-related recalculations and access expansion. See Salesforce Data Skew guidance and Salesforce Sharing Overview.

QUESTION NO: 3

Universal Containers stores financial transaction data in Salesforce. Compliance requires UC to retain a history of changes to selected transaction fields for 10 years, including the user who made the change, the date of the change, and the previous and new values.

Which two features should a data architect recommend?

Choose 2 answers.

A. Enable Field History Tracking.

B. Use Field Audit Trail.

C. Use Setup Audit Trail to retain transaction-field values.

D. Create a workflow email alert whenever a transaction field changes.

ANSWER: A B

Explanation:

Enable Field History Tracking captures field-value changes, including the user, date and time, previous value, and new value for configured fields. It provides the underlying audit history required for selected objects and fields.

Use Field Audit Trail extends retention of field history beyond the standard retention period and supports a policy-based approach for retaining historical records for the required compliance duration. Field Audit Trail is part of Salesforce Shield.

The architect should identify the exact fields subject to audit requirements, establish retention policies, and test access to retained history records. Salesforce documents Field History Tracking and Field Audit Trail.

QUESTION NO: 4

DreamHouse Realty has a legacy system that captures Branch Offices and Transactions.

DreamHouse Realty has 15 Branch Offices. Transactions can relate to any Branch Office.

DreamHouse Realty has created hundreds of thousands of Transactions per year.

A Data Architect needs to denormalize this data model into a single Transaction object with a Branch Office picklist.

What are two important considerations for the Data Architect in this scenario? (Choose two.)

A. Standard list view in-line editing.

B. Limitations on Org data storage.

C. Bulk API limitations on picklist fields.

D. Limitations on master-detail relationships.

ANSWER: B C

Explanation:

Limitations on Org data storage. is an important consideration because the solution will retain hundreds of thousands of Transaction records each year in Salesforce. The architect must size projected record and file storage, account for retention requirements, and establish an archival or externalization strategy if growth will exceed the organization's purchased capacity. Although a Branch Office picklist has only a small number of values, denormalizing changes the long-term data footprint and removes a related-object design that might otherwise centralize branch attributes. Salesforce documents both record-storage allocation and the need to monitor consumption as data volumes grow. See Salesforce Data Storage Allocations.

Bulk API limitations on picklist fields. is also important because the legacy system produces high transaction volumes and bulk loading will be required. Every imported Branch Office value must match a configured picklist value, particularly when the picklist is restricted; invalid or unavailable values cause row-level load errors. The integration must therefore map source branch values consistently, deploy picklist metadata before loading Transactions, and handle failed rows through an error-management process. Salesforce's bulk-processing guidance and picklist documentation describe how data loads operate at scale and how restricted picklists enforce valid values. See Bulk API Developer Guide.

QUESTION NO: 5

Universal Container require all customers to provide either a phone number of an email address when registering for an account.

What should the data architect use to ensure this requirement is met?

A. validation Rule

B. required Fields

C. Apex Class

D. Process Builder

ANSWER: A

Explanation:

A validation Rule is the appropriate declarative control because the requirement is conditional across two fields: an account registration is valid when at least one of Phone or Email is populated. A validation rule can evaluate both fields whenever the record is created or edited and prevent the record from being saved only when both are blank. For example, a rule formula can use `AND(ISBLANK(Phone), ISBLANK(Email__c))`, adjusted for the organization's actual email field API name. When this formula evaluates to true, Salesforce displays a clear error message and requires the user or integration to supply a value in either field before continuing. This approach preserves the intended flexibility: neither field is individually mandatory, but the record cannot be saved without at least one contact method. Validation rules are enforced consistently through the user interface, APIs, imports, and integrations, helping maintain data quality regardless of how account records enter Salesforce. Salesforce describes validation rules and their save-blocking behavior in its Validation Rules overview. Formula functions such as `ISBLANK` are documented in the Salesforce Formula Operators and Functions reference.

QUESTION NO: 6

NTO has outgrown its current salesforce org and will be migrating to new org shortly. As part of this process NTO will be migrating all of its metadata and data. NTO's data model in the source org has a complex relationship hierarchy with several master detail and lookup relationships across objects, which should be maintained in target org.

What 3 things should a data architect do to maintain the relationship hierarchy during migration?

Choose 3 answers:

A. Use data loader to export the data from source org and then import or Upsert into the target org in sequential order.

B. Create a external id field for each object in the target org and map source record ID's to this field.

C. Redefine the master detail relationship fields to lookup relationship fields in the target org.

D. Replace source record ID's with new record ID's from the target org in the import file.

E. Keep the relationship fields populated with the source record ID's in the import file.

ANSWER: A B D

Explanation:

To preserve a complex relationship hierarchy, NTO must establish a reliable way to translate source-record identifiers to the newly generated identifiers in the target org. Creating an external ID field for each migrated object provides a stable, unique business or migration key that can be used for upserts and relationship resolution. Salesforce supports populating relationship fields by referencing a parent record's external ID, avoiding dependency on Salesforce record IDs that differ between orgs.

The migration must also be sequenced according to dependencies. Parent records must exist before child records can be inserted or upserted with required master-detail fields and other relationship references. Using Data Loader to export data and then load objects in this dependency-aware order supports this process; for example, top-level parent objects are loaded before their related child objects.

When import files contain direct relationship-ID values rather than external-ID relationship references, the source IDs must be replaced with the corresponding target IDs after parent records are loaded. Maintaining a source-to-target ID cross-reference enables those substitutions and ensures lookup and master-detail fields point to the intended target records. Salesforce documents Data Loader's import and upsert behavior at Data Loader Guide and external IDs at Custom Field Attributes.

QUESTION NO: 7

Universal Container has implemented Sales Cloud to manage patient and related health records. During a recent security audit of the system it was discovered that same standard and custom fields need to encrypted.

Which solution should a data architect recommend to encrypt existing fields?

A. Use Apex Crypto Class encrypt customer and standard fields.

B. Implement classic encryption to encrypt custom and standard fields.

C. Implement Shield Platform Encryption to encrypt custom and standard fields.

D. Expert data out of Salesforce and encrypt custom and standard fields.

ANSWER: C

Explanation:

Implement Shield Platform Encryption to encrypt custom and standard fields. Salesforce Shield Platform Encryption is designed to protect sensitive data at rest while allowing organizations to continue using Salesforce applications and standard platform capabilities. It supports encryption for eligible standard fields and custom fields, making it appropriate when a security audit identifies protected health or patient-related data across both field types. After encryption is configured and the relevant fields are enabled, Salesforce encrypts existing field values as part of its encryption process, so this approach addresses data already stored in the organization as well as data entered afterward.

For health-record scenarios, Platform Encryption also provides a key-management model through the Salesforce key hierarchy, including tenant secrets and options such as customer-managed keys and Bring Your Own Key where licensing and requirements call for them. A data architect should first assess field eligibility and the effect encryption can have on features such as filtering, integrations, reporting, and search, then plan deployment and validation accordingly. Salesforce documents supported field types, feature considerations, and key-management capabilities in its Shield Platform Encryption overview and Platform Encryption implementation guidance.

QUESTION NO: 8

Universal Containers (UC) has a requirement to create an Account plan object that is related to the Account object. Each Account plan needs to have an Account object, but the accessibility requirement of the Account plan is different from the Account object. What should an Architect recommend?

A. Create a custom account plan object as detail with Account as mater in a master-detail relationship.

B. Create a custom account plan object as detail with Account as master with additional sharing rules to allow access.

C. Create an account plan object with a lookup relations to Account without any validation rules to enforce the Account association.

D. Create an account plan object with a lookup relationship to Account with validation rules to enforce the Account association.

ANSWER: D

Explanation:

Create an account plan object with a lookup relationship to Account with validation rules to enforce the Account association. A lookup relationship is appropriate because it allows the Account plan custom object to retain its own sharing and security model rather than inheriting access from the related Account. This supports the stated need for Account plans to have different accessibility requirements while still providing a direct relationship to the Account record.

The Account association can be made mandatory at the data-model level by configuring the lookup field as required, or enforced through a validation rule when the business requirement needs conditional logic or when the relationship must be checked in conjunction with other field values. This design separates relationship integrity from record access: every Account plan remains associated with an Account, while organization-wide defaults, role hierarchy behavior, sharing rules, teams, or Apex managed sharing can be designed specifically for the Account plan object.

This approach follows Salesforce's distinction between lookup relationships, which do not automatically inherit parent access, and master-detail relationships, which tightly couple ownership and sharing to the parent. See Salesforce's relationship considerations documentation and data access and sharing documentation.

QUESTION NO: 9

An Architect needs to document the data architecture for a multi-system, enterprise

Salesforce implementation.

Which two key artifacts should the Architect use? (Choose two.)

A. User stories

B. Data model

C. Integration specification

D. Non-functional requirements

ANSWER: B C

Explanation:

Data model and **Integration specification** are the core artifacts for documenting data architecture across an enterprise Salesforce landscape. A data model documents the business entities, attributes, relationships, ownership, and data constraints that define how information is represented within Salesforce. This gives stakeholders and delivery teams a shared, durable view of the system of record, required data structures, and how data supports business processes. In Salesforce, object relationships and field design are fundamental to maintaining usable, scalable data structures.

An integration specification complements the data model by documenting how data moves between Salesforce and external systems. It should identify participating systems, source and target data, mappings, interfaces or APIs, synchronization

direction and frequency, transformation rules, error handling, and security considerations. In a multi-system implementation, these details establish clear data ownership and ensure that integrations preserve data quality and consistency as records cross system boundaries. Together, these artifacts describe both the structure of enterprise data and its movement across the architecture. Salesforce guidance on data modeling and integration architecture supports using these designs to plan scalable, connected solutions.

QUESTION NO: 10

UC is building a salesforce application to track contacts and their respective conferences that they have attended with the following requirements:

1.Contacts will be stored in the standard contact object.

2.Conferences will be stored in a custom conference__c object.

3.Each contact may attend multiple conferences and each conference may be related to multiple contacts.

How should a data architect model the relationship between the contact and conference objects?

A. Implement a Contact Conference junction object with master detail relationship to both contact and conference__c.

B. Create a master detail relationship field on the Contact object.

C. Create a master detail relationship field on the Conference object.

D. Create a lookup relationship field on contact object.

ANSWER: A

Explanation:

Implement a Contact Conference junction object with master detail relationship to both contact and conference__c. This is the standard Salesforce data-modeling pattern for a many-to-many relationship. The Contact Conference custom object represents a single attendance record, linking one Contact to one Conference__c record. Because multiple Contact Conference records can be associated with the same Contact, a contact can be connected to any number of conferences. Similarly, because multiple junction records can point to the same Conference__c record, one conference can include any number of contacts.

A junction object also provides a scalable place to store attributes specific to the person's attendance rather than to either parent record alone. Examples include registration date, attendance status, check-in time, ticket type, speaker role, or post-conference feedback. Master-detail relationships are appropriate when each attendance record is dependent on and owned by both its parent Contact and Conference__c records. Salesforce describes many-to-many modeling through a custom junction object with two master-detail relationships in its relationship guidance. See Salesforce Trailhead: Data Modeling and Salesforce Help: Many-to-Many Relationships.

QUESTION NO: 11

Developers at Universal Containers need to build a report for the business which displays Accounts opened in the past year grouped by industry. This report will also include information from contacts, opportunities, and orders. There are several million Accounts in the system. Which two options should be recommended to make this report perform well and satisfy the business need?

A. Use triggers to populate denormalized related fields on the Account.

B. Use an indexed data field with bounded data filters.

C. Use Formula fields to surface information I related entities on the report.

D. Use unbounded date ranges to filter the report.

ANSWER: A B

Explanation:

“Use triggers to populate denormalized related fields on the Account” is appropriate because the report spans a very large Account population and requires data originating from several related objects. Maintaining the needed reporting values on the Account record allows the report to use a simpler, Account-centered data set rather than repeatedly traversing multiple relationships at runtime. A trigger or other automation can maintain fields such as relevant totals, status indicators, or summary values when Contacts, Opportunities, or Orders change. This approach is especially useful when the business accepts reporting on precomputed values and needs consistently responsive report execution at scale.

“Use an indexed data field with bounded data filters” is also essential. The report should constrain the Account population to a specific start and end date representing the past year, using an indexed field where possible. Standard fields such as CreatedDate are indexed, and a selective, bounded filter enables Salesforce to identify the qualifying subset instead of processing millions of Account records. Grouping that reduced result set by Industry then satisfies the requested analysis while minimizing report-processing cost. Salesforce describes selective filtering and indexed fields as key factors in efficient query execution; the same principle is fundamental when designing scalable report data access. See Salesforce Query Selectivity and Salesforce Report Types.

QUESTION NO: 12

Universal container (UC) would like to build a Human resources application on Salesforce to manage employee details, payroll, and hiring efforts. To adequately and store the relevant data, the application will need to leverage 45 custom objects. In addition to this, UC expects roughly 20,00 API calls into Salesfoce from an n-premises application daily.

Which license type should a data architect recommend that best fits these requirements?

A. Service Cloud

B. Lightning platform Start

C. Lightning Platform plus

D. Lightning External Apps Starts

ANSWER: C

Explanation:

Lightning Platform plus is the appropriate recommendation because this is a custom, internal business application with substantial data-model and integration requirements. The HR solution needs 45 custom objects for employee, payroll, recruiting, and related records. Lightning Platform Plus is designed for organizations building custom applications on the Salesforce Platform and supports up to 110 custom objects, providing enough capacity for this data model and room for future expansion.

It also fits the anticipated integration volume. Salesforce API request allocations are determined in part by the organization’s edition and applicable user-license allocations, and Lightning Platform Plus has a higher per-user API entitlement than the entry-level Platform offering. This makes it the suitable Platform license family where an on-premises application is expected to make approximately 20,000 API calls each day. The architect should still validate the organization’s total API limit using Salesforce’s API Request Limits reporting and calculate the required number of licenses against the final integration design, retry behavior, and other API consumers.

Salesforce documents Platform license capabilities and object entitlements in its license type documentation. API allocation and monitoring guidance is available in the Salesforce API limits documentation.

QUESTION NO: 13

Universal Containers wants to implement a data -quality process to monitor the data that users are manually entering into the system through the Salesforce UI. Which approach should the architect recommend?

A. Allow users to import their data using the Salesforce Import tools.

B. Utilize a 3rd -party solution from the AppExchange for data uploads.

C. Utilize an app from the AppExchange to create data -quality dashboards.

D. Use Apex to validate the format of phone numbers and postal codes.

ANSWER: C

Explanation:

Utilize an app from the AppExchange to create data -quality dashboards. A dashboard-oriented data-quality solution is appropriate because the requirement is to *monitor* data entered through the Salesforce user interface across the organization. Such a solution can provide ongoing visibility into quality measures such as completeness, consistency, duplicate risk, validity, and exception trends. It enables data stewards and business owners to identify where quality is deteriorating, focus remediation work on the most significant issues, and track whether corrective actions improve data over time. This supports a repeatable data-governance process rather than addressing only isolated fields or records. Salesforce dashboards are designed to visually present report data and can be used to track operational metrics; an AppExchange product can extend that capability with data-quality-specific profiling, scoring, and monitoring features. Salesforce also positions AppExchange as the marketplace for installing applications that extend Salesforce functionality, allowing an architect to select a purpose-built capability without developing and maintaining a custom monitoring solution. See Salesforce guidance on dashboards and the Salesforce AppExchange.

QUESTION NO: 14

Universal Container is using Salesforce for Opportunity management and enterprise resource planning (ERP) for order management. Sales reps do not have access to the ERP and have no visibility into order status.

What solution a data architect recommend to give the sales team visibility into order status?

A. Leverage Canvas to bring the order management UI in to the Salesforce tab.

B. Build batch jobs to push order line items to salesforce.

C. leverage Salesforce Connect top bring the order line item from the legacy system to Salesforce.

D. Build real-time integration to pull order line items into Salesforce when viewing orders.

ANSWER: C

Explanation:

“leverage Salesforce Connect top bring the order line item from the legacy system to Salesforce.” is the appropriate recommendation because Salesforce Connect gives Salesforce users real-time access to data that remains in an external system, such as an ERP. The ERP remains the system of record for orders and their statuses, while Salesforce exposes the external order data through external objects. Sales representatives can then view current order information in Salesforce without receiving ERP credentials or requiring a copied, independently maintained order-data store in Salesforce.

This approach is particularly well suited when order status must remain current and the organization wants to avoid the storage, synchronization, latency, and reconciliation concerns that come with replicating potentially large order and order-line datasets. A data architect can configure an external data source and use the appropriate adapter—commonly OData or a custom adapter—to connect Salesforce to the ERP. External objects can also be related to Salesforce records, allowing order information to be presented in the relevant account or opportunity context. Salesforce Connect is designed specifically for securely accessing external data on demand while it remains outside the Salesforce org. See Salesforce: External Objects and Salesforce: Salesforce Connect.

QUESTION NO: 15

A company has 12 million records, and a nightly integration queries these records.

Which two areas should a Data Architect investigate during troubleshooting if queries are timing out? (Choose two.)

A. Make sure the query doesn't contain NULL in any filter criteria.

B. Create a formula field instead of having multiple filter criteria.

C. Create custom indexes on the fields used in the filter criteria.

D. Modify the integration users' profile to have View All Data.

ANSWER: A C

Explanation:

With 12 million records, query selectivity and index use are central to preventing timeouts. “Make sure the query doesn't contain NULL in any filter criteria.” is an important investigation area because Salesforce indexes generally do not include records where the indexed field is null. A filter such as `Some_Field__c = null` can therefore require scanning a very large portion of the table, making the query nonselective and substantially increasing execution time. The architect should review whether null filtering is necessary and whether the data model or query can be adjusted to use a selective, indexed predicate.

“Create custom indexes on the fields used in the filter criteria.” is also appropriate when the integration filters on fields that are not already indexed. Custom indexes can enable Salesforce to narrow the candidate record set efficiently rather than scanning millions of rows. Index requests should be based on the actual query predicates and expected selectivity; reviewing query plans helps determine whether a filter can use an index and whether it meets Salesforce’s selectivity thresholds. Salesforce documents indexed fields, custom indexes, and query optimization guidance in its SOQL and SOSL Reference: Custom Indexes and SOQL Selectivity and Query Optimization.

QUESTION NO: 16

Cloud Kicks has the following requirements:

• Their Shipment custom object must always relate to a Product, a Sender, and a Receiver

(all separate custom objects).

• If a Shipment is currently associated with a Product, Sender, or Receiver, deletion ofthose records should not be allowed.

• Each custom object must have separate sharing models.

What should an Architect do to fulfill these requirements?

A. Associate the Shipment to each parent record by using a VLOOKUP formula field.

B. Create a required Lookup relationship to each of the three parent records.

C. Create a Master-Detail relationship to each of the three parent records.

D. Create two Master-Detail and one Lookup relationship to the parent records.

ANSWER: B

Explanation:

Create a required Lookup relationship to each of the three parent records. This design makes Product, Sender, and Receiver mandatory references on every Shipment, so users cannot save a Shipment unless all three related records are populated. Lookup relationships are appropriate because the related custom objects must retain independent ownership and sharing behavior. Unlike a master-detail relationship, a lookup relationship does not cause the child record to inherit the parent record’s owner or sharing settings; therefore, Shipment, Product, Sender, and Receiver can each use their own organization-wide defaults, sharing rules, and access model.

For required lookup relationships, Salesforce prevents deletion of a referenced lookup record while child records still depend on it. This satisfies the requirement that a Product, Sender, or Receiver cannot be deleted when an associated Shipment exists, while preserving separate sharing models. The architect should create one required lookup field from Shipment to each of the three distinct custom objects and use the lookup deletion behavior that restricts deletion of referenced records. Salesforce documents lookup relationships and their behavior in the Relationship Considerations guidance and explains how record access is controlled independently through Salesforce Sharing.

QUESTION NO: 17

NTO uses salesforce to manage relationships and track sales opportunities. It has 10 million customers and 100 million opportunities. The CEO has been complaining 10

minutes to run and sometimes failed to load, throwing a time out error.

Which 3 options should help improve the dashboard performance?

Choose 3 answers:

A. Use selective queries to reduce the amount of data being returned.

B. De-normalize the data by reducing the number of joins.

C. Remove widgets from the dashboard to reduce the number of graphics loaded.

D. Run the dashboard for CEO and send it via email.

E. Reduce the amount of data queried by archiving unused opportunity records.

ANSWER: A C E

Explanation:

“Use selective queries to reduce the amount of data being returned.” is appropriate because each dashboard component is backed by a report query. Filters that substantially narrow the records evaluated, especially on indexed fields where applicable, reduce query cost and help reports complete within platform limits. Salesforce’s guidance for large data volumes emphasizes selective filtering as a key factor in query performance.

“Remove widgets from the dashboard to reduce the number of graphics loaded.” improves performance because every dashboard component requires report execution and rendering. Reducing the number of components lowers the aggregate reporting work, response payload, and browser rendering effort required when the CEO opens the dashboard.

“Reduce the amount of data queried by archiving unused opportunity records.” is also effective for an organization with 100 million opportunities. Moving obsolete opportunities out of the operational dataset reduces the volume that active reports must search and aggregate, provided reporting filters and retention requirements are designed accordingly. An archive strategy is a standard large-data-volume practice that keeps frequently accessed operational data manageable while retaining historical information elsewhere. See Salesforce guidance on selective queries and SOQL performance and data archiving.

QUESTION NO: 18

Universal Containers is creating a new B2C service offering for consumers to ship goods across continents. This is in addition to their well-established B2B offering. Their current Salesforce org uses the standard Account object to track B2B customers. They are expecting to have over 50,000,000 consumers over the next five years across their 50 business regions. B2C customers will be individuals. Household data is not required to be stored. What is the recommended data model for consumer account data to be stored in

Salesforce?

A. Use the Account object with Person Accounts and a new B2C page layout.

B. Use the Account object with a newly created Record Type for B2C customers.

C. Create a new picklist value for B2C customers on the Account Type field.

D. Use 50 umbrella Accounts for each region, with customers as associated Contacts.

ANSWER: A

Explanation:

Use the Account object with Person Accounts and a new B2C page layout. Person Accounts are Salesforce's standard data model for an individual consumer who is both the customer account and the person with whom the company interacts. A Person Account combines account-level information with the person/contact attributes needed for an individual, while retaining compatibility with the existing Account-based B2B model in the same org. This lets Universal Containers continue to represent business customers as business Accounts while representing consumers as Person Accounts, using an appropriate record type and tailored page layout for the B2C experience.

The lack of a household-data requirement is important: there is no need to introduce a household relationship model or make an artificial parent account structure simply to represent individuals. Person Accounts directly model each consumer as an independent customer, which supports consumer-specific fields, activity management, service processes, reporting, sharing, and integrations using Salesforce's supported account and contact semantics. The anticipated consumer volume requires sound overall data-volume architecture, indexing, retention, and integration design, but it does not change the fundamental recommendation to use Salesforce's purpose-built individual-account model. Review the Person Account object documentation at Salesforce Developer Documentation and Salesforce's overview of account and contact relationships at Trailhead.

QUESTION NO: 19

Universal Containers (UC) has adopted Salesforce as its primary sales automated tool. UC has 100,00 customers with a growth rate of 10% a year, UC uses an on-premise webbased billing and invoice system that generates over 1 million invoices a year supporting a monthly billing cycle.

The UC sales team needs to be able to pull a customer record and view their account status, Invoice history, and opportunities without navigating outside of Salesforce.

What should a data architect use to provide the sales team with the required functionality?

A. Create a custom object and migrate the last 12 months of Invoice data into Salesforce so it can be displayed on the Account layout.

B. Write an Apex callout and populate a related list to display on the account record.

C. Create a mashup page that will present the billing system records within Salesforce.

D. Create a visual force tab with the billing system encapsulated within an iframe.

ANSWER: C

Explanation:

Create a mashup page that will present the billing system records within Salesforce. is the appropriate solution because invoice and billing data is high-volume, originates in an on-premises system, and must remain available to sales users in the context of the Salesforce customer record. A mashup presents relevant information from an external application in the Salesforce user experience, allowing users to see current account status and invoice history without leaving Salesforce. Opportunities remain available from Salesforce's native data model alongside the externally sourced billing information.

This approach avoids unnecessarily copying a rapidly growing invoice dataset into Salesforce merely for display. Instead, the billing system remains the system of record, while the Salesforce interface supplies the sales team with a unified customer view. A properly designed mashup can pass customer context to the billing application and use appropriate authentication, authorization, and integration controls so that the displayed records are specific to the account being viewed. This aligns with Salesforce integration architecture guidance to select a pattern based on data ownership, volume, latency, and user-experience requirements. See Salesforce's integration patterns overview and its external data guidance.

QUESTION NO: 20

Universal Containers (UC) management has identified a total of ten text fields on the Contact object as important to capture any changes made to these fields, such as who made the change, when they made the change, what is the old value, and what is the new value. UC needs to be able to report on these field data changes within Salesforce for the past 3 months. What are two approaches that will meet this requirement? Choose 2 answers

A. Create a workflow to evaluate the rule when a record is created and use field update actions to store previous values for these ten fields in ten new fields.

B. Write an Apex trigger on Contact after insert event and after update events and store the old values in another custom object.

C. Turn on field Contact object history tracking for these ten fields, then create reports on contact history.

D. Create a Contact report including these ten fields and Salesforce Id, then schedule the report to run once a day and send email to the admin.

ANSWER: B C

Explanation:

Turning on field Contact object history tracking for these ten fields directly satisfies the audit and reporting requirement. Salesforce field history tracking records the user who made the change, the date and time of the change, and the prior and new values. Contact supports tracking for up to 20 fields, so all ten identified text fields fit within the standard limit. The resulting Contact History data can be used in Salesforce reports, and the requested three-month reporting period is well within the standard field-history retention period. Salesforce documents field history capabilities and retention at Field History Tracking.

Writing an Apex trigger on Contact after insert event and after update events and storing the old values in another custom object is also a viable custom audit approach. On updates, Apex can compare the old and new Contact field values, then create audit records containing the changed field, old value, new value, changing user, and timestamp. Because these records are stored in a custom object, UC can define a custom report type and report on the audit entries for any required date range, including the past three months. This approach also provides flexibility for retention, reporting fields, and audit-object design. Salesforce describes access to prior values in trigger context through Apex Trigger Context Variables.

QUESTION NO: 21

Get Cloudy Consulting monitors 15,000 servers, and these servers automatically record their status every 10 minutes. Because of company policy, these status reports must be maintained for 5 years. Managers at Get Cloudy Consulting need access to up to one week's worth of these status reports with all of their details.

An Architect is recommending what data should be integrated into Salesforce and for how long it should be stored in Salesforce.

Which two limits should the Architect be aware of? (Choose two.)

A. Data storage limits

B. Workflow rule limits

C. API Request limits

D. Webservice callout limits

ANSWER: A C

Explanation:

Data storage limits are critical because the source produces 15,000 status reports every 10 minutes: 2.16 million reports per day and more than 15 million reports for a seven-day online window. If records and their detailed fields are retained in standard Salesforce objects, this volume can consume allocated data storage rapidly. The architecture should therefore calculate storage from expected record size and growth, retain only the detail managers require in Salesforce, and consider an external system of record with summarized or selectively replicated data in Salesforce. Salesforce documents organization storage usage and allocation considerations at Salesforce Help: Data Storage.

API Request limits also directly affect the design because automated status ingestion requires a very large and sustained volume of requests. The integration must account for the organization's API request entitlement, select an efficient ingestion pattern such as Bulk API for high-volume loads, batch records appropriately, and monitor usage so normal business integrations remain available. Salesforce applies API request limits over rolling time windows, and an integration can be

blocked or delayed when it exceeds its available allocation. The applicable limit model and monitoring guidance are described in Salesforce Developer Documentation: API Request Limits and Allocations.

QUESTION NO: 22

A company wants to document the data architecture of a Salesforce organization.

What are two valid metadata types that should be included? (Choose two.)

A. RecordType

B. Document

C. CustomField

D. SecuritySettings

ANSWER: A C

Explanation:

RecordType and **CustomField** are valid Salesforce metadata types and are central to documenting an organization's data architecture. A CustomField definition captures the structure of data stored on a standard or custom object, including its field type, length or precision, requiredness, default value, formula behavior where applicable, help text, and relationship characteristics. Capturing these definitions provides the detailed field-level view needed for a data model and data dictionary.

RecordType metadata documents the business-specific variations in how records for an object are categorized and managed. Record types can determine the available picklist values, the page layout presented to users, and the business processes applied for applicable objects. Including them in architecture documentation helps explain how one underlying object supports distinct business processes, regions, product lines, or customer segments without creating separate objects.

Together, these metadata types describe both the stored data structure and important contextual variations in how that data is classified and presented. Salesforce's Metadata API reference identifies these as supported metadata types: Metadata Coverage Report. Salesforce also describes the role of record types in tailoring business processes and picklist values in Record Types.

QUESTION NO: 23

Universal Container is Implementing salesforce and needs to migrate data from two legacy systems. UC would like to clean and duplicate data before migrate to Salesforce.

Which solution should a data architect recommend a clean migration?

A. Define external IDs for an object, migrate second database to first database, and load into Salesforce.

B. Define duplicate rules in Salesforce, and load data into Salesforce from both databases.

C. Set up staging data base, and define external IDs to merge, clean duplicate data, and load into Salesforce.

D. Define external IDs for an object, Insert data from one database, and use upsert for a second database

ANSWER: C

Explanation:

Set up staging data base, and define external IDs to merge, clean duplicate data, and load into Salesforce. is the appropriate approach because a staging database provides a controlled area to consolidate extracts from both legacy systems before any production Salesforce records are created. The migration team can profile the incoming data, standardize formats and values, apply matching logic across both sources, identify duplicate records, and apply defined survivorship rules to create one trusted version of each record. This makes data quality decisions repeatable, auditable, and testable during migration rehearsals. External IDs should be retained or created for the legacy source identifiers and the consolidated master records.

They enable reliable relationship mapping during load, support idempotent upsert operations where appropriate, and make reconciliation between legacy records and Salesforce records substantially easier. Salesforce's data-import guidance emphasizes preparing, cleansing, and mapping data before import, while external IDs are designed to identify records from external systems and support integration and upsert matching. See Salesforce Help: Import Data Into Salesforce and Salesforce Developer Documentation: Field Types and External IDs.

QUESTION NO: 24

UC is preparing to implement sales cloud and would like to its users to have read only access to an account record if they have access to its child opportunity record. How would a data architect implement this sharing requirement between objects?

A. Create a criteria-based sharing rule.

B. Implicit sharing will automatically handle with standard functionality.

C. Add appropriate users to the account team.

D. Create an owner-based sharing rule.

ANSWER: B

Explanation:

Implicit sharing will automatically handle with standard functionality. Salesforce provides parent implicit sharing to support the natural relationship between parent and child records. When a user is granted access to an Opportunity, Salesforce automatically grants that user read-only access to the parent Account when needed. This behavior lets the user view the account context associated with the opportunity while preserving the intended access level on the Account itself. It is evaluated by the platform as part of its standard record-sharing model, so the architect does not need to maintain custom sharing records or create rules specifically to expose the parent Account for each shared Opportunity. This is particularly useful in Sales Cloud because Opportunities are children of Accounts and users commonly need account details, such as account name and related information, while working with an opportunity. The access granted through parent implicit sharing is read-only and follows Salesforce's built-in sharing behavior; it does not make the user an Account owner or grant edit rights to the Account. Salesforce documents this as implicit sharing, including the automatic access relationship between child records and their parents. See Salesforce Help: Implicit Sharing and Salesforce Security Implementation Guide: Data Access and Sharing.