

Salesforce Developer Interview Mega Guide (500 Q&A)

1. Apex Fundamentals & OOP Concepts (50 Q&A)

- Core Apex syntax
 - OOP (inheritance, polymorphism, encapsulation)
 - Difference between Apex & Java
 - Collections (List, Set, Map)
 - Exception handling in Apex
-

2. Triggers (40 Q&A)

- Trigger events (before/after insert/update/delete)
 - Bulkification and best practices
 - Recursive trigger prevention
 - Trigger framework design pattern
 - Use cases like "auto-assigning Account Manager on new Opportunity"
-

3. Asynchronous Apex (40 Q&A)

- Future methods vs Queueable vs Batch vs Schedulable
 - Use cases for each
 - Limits & governor restrictions
 - Chaining batch jobs
 - Queueable job monitoring
-

4. LWC (Lightning Web Components) (60 Q&A)

- Lifecycle hooks
- @track, @wire, @api decorators

- Calling Apex methods from LWC
 - Communication patterns (parent-child, pub-sub, LMS)
 - LWC performance optimization
-

5. Aura Components & Migration (20 Q&A)

- Aura vs LWC
 - Aura event handling
 - Migration best practices
-

6. Visualforce (15 Q&A)

- When to use VF today
 - Integration with Lightning pages
 - VF + Apex controller
-

7. SOQL & SOSL (30 Q&A)

- Relationship queries
 - Semi-join, anti-join
 - Aggregate queries
 - Avoiding governor limits in SOQL
 - SOSL vs SOQL
-

8. Integration (40 Q&A)

- REST vs SOAP in Salesforce
- Named Credentials
- Platform Events
- Change Data Capture (CDC)

- Outbound messages
 - Authentication (OAuth, JWT)
-

9. OmniStudio (30 Q&A)

- OmniScripts basics
 - DataRaptor Extract/Transform/Load
 - FlexCards
 - Integration Procedures
-

10. Sales Cloud (25 Q&A)

- Lead conversion process
 - Opportunity stages & forecasting
 - Territory management
-

11. Service Cloud (25 Q&A)

- Case management
 - Omni-channel routing
 - Knowledge management
 - Entitlements
-

12. Experience Cloud (25 Q&A)

- User authentication
 - Guest user access
 - Sharing rules in portals
 - Branding & customization
-

13. Security & Sharing (30 Q&A)

- OWD, Role hierarchy
 - Sharing rules
 - Apex managed sharing
 - With/without sharing keywords
-

14. Deployment & DevOps (30 Q&A)

- Change sets vs SFDX
 - CI/CD pipelines
 - Git branching strategies
 - Ant migration tool
-

15. Governor Limits & Performance (25 Q&A)

- SOQL per transaction limit
 - Heap size & CPU time limits
 - Best practices to avoid hitting limits
-

16. Testing (25 Q&A)

- @isTest, Test.startTest/stopTest
 - Test data creation best practices
 - Mocking callouts (HttpCalloutMock)
 - Code coverage requirements
-

17. Design Patterns (20 Q&A)

- Trigger handler pattern
- Singleton pattern

- Strategy pattern for custom logic
-

18. Advanced Topics (30 Q&A)

- Custom Metadata vs Custom Settings
 - Dynamic Apex (Schema class)
 - Tooling API vs Metadata API
 - Lightning Data Service
-

19. Real-Time Scenarios (100+ Q&A)

◆ Part 1: Apex Fundamentals & OOP Concepts (50 Q&A)

1. What is Apex?

- **Answer:** Apex is a strongly typed, object-oriented programming language used to execute flow and transaction control statements on Salesforce servers.
 - **Explanation:** It's like Java but runs natively on the Salesforce platform and respects governor limits.
-

2. How is Apex similar to Java?

- **Answer:** Syntax, OOP principles (inheritance, polymorphism), exception handling, and data types are similar.
 - **Explanation:** Developers from Java background can easily adapt to Apex.
-

3. Difference between Apex and Java?

- **Answer:** Apex is cloud-hosted, has governor limits, and is tightly coupled with Salesforce objects, while Java is general-purpose and runs on JVM.
- **Example:** In Apex, `Account a = new Account(Name='Test');` directly maps to Salesforce DB.

4. What are collections in Apex?

- **Answer:** Collections are types that hold multiple values: List, Set, Map.
 - **Example:** `List<String> names = new List<String>();`
-

5. Difference between List, Set, and Map?

- **Answer:**
 - **List:** Ordered, allows duplicates
 - **Set:** Unordered, no duplicates
 - **Map:** Key-value pairs
 - **Example:** `Map<Id, Account> accMap = new Map<Id, Account>();`
-

6. What is a sObject?

- **Answer:** A generic data type that represents Salesforce records (standard/custom objects).
 - **Example:** `Account acc = new Account();`
-

7. How do you create a new record in Apex?

- **Answer:** Using insert DML.
 - **Example:**
 - `Account acc = new Account(Name='Test');`
 - `insert acc;`
-

8. What is DML in Apex?

- **Answer:** DML (Data Manipulation Language) is used to insert, update, delete, merge, undelete, upsert records in Salesforce.
-

9. What is SOQL?

- **Answer:** Salesforce Object Query Language, used to query data from Salesforce objects.
 - **Example:** `SELECT Id, Name FROM Account WHERE Industry='IT'`
-

10. What is SOSL?

- **Answer:** Salesforce Object Search Language, used to search across multiple objects and fields.
 - **Example:**
 - `FIND 'Test' IN ALL FIELDS RETURNING Account(Id, Name), Contact(Id, Name)`
-

11. What are Governor Limits?

- **Answer:** Salesforce-imposed runtime limits (like max SOQL queries, heap size, DML) to ensure efficient multi-tenant performance.
-

12. Example of a governor limit?

- **Answer:** Max 100 SOQL queries per transaction.
 - **Explanation:** To enforce bulkification.
-

13. What is a Static variable?

- **Answer:** A variable tied to the class instead of instances; retains value for transaction.
 - **Example:** `public static Boolean hasRun = false;`
-

14. Difference between Static and Final?

- **Answer:**
 - **Static:** Class-level scope.

- **Final:** Value cannot be changed after initialization.
-

15. What are Access Modifiers in Apex?

- **Answer:** public, private, protected, global determine visibility of classes/methods/variables.
-

16. What is the use of the with sharing keyword?

- **Answer:** Enforces sharing rules (record-level security) in Apex classes.
-

17. Difference between with sharing and without sharing?

- **Answer:**
 - **with sharing:** Respects user's record-level access.
 - **without sharing:** Ignores record access restrictions.
-

18. What is Inheritance in Apex?

- **Answer:** Mechanism where one class derives properties/methods of another.
 - **Example:** class Child extends Parent { }
-

19. What is Polymorphism in Apex?

- **Answer:** Same method name but different implementations.
 - **Example:** Method overriding in child classes.
-

20. What is Encapsulation?

- **Answer:** Wrapping data and methods together, controlling access with access modifiers.
-

21. Difference between Abstract and Virtual class?

- **Answer:**
 - **Abstract:** Cannot be instantiated, may contain abstract methods.
 - **Virtual:** Can be instantiated, but methods can be overridden.
-

22. What is an Interface in Apex?

- **Answer:** A contract that defines methods without implementation.
 - **Example:** `interface MyInterface { void doSomething(); }`
-

23. Difference between Interface and Abstract class?

- **Answer:**
 - **Interface:** Only method signatures.
 - **Abstract:** Can have both abstract and concrete methods.
-

24. What is a Constructor in Apex?

- **Answer:** A special method called automatically when an object is created.
-

25. Can we overload constructors in Apex?

- **Answer:** Yes, multiple constructors can exist with different parameters.
-

26. What is Exception Handling in Apex?

- **Answer:** Handling runtime errors using try-catch-finally blocks.
-

27. What is a Custom Exception?

- **Answer:** User-defined exception extending Exception class.
- **Example:**

- `class MyException extends Exception {}`
-

28. What is a Null Pointer Exception?

- **Answer:** Occurs when accessing a property or method of a null object.
-

29. What is the difference between Primitive and Non-Primitive data types?

- **Answer:**
 - **Primitive:** int, boolean, String, Double.
 - **Non-Primitive:** sObjects, collections, custom classes.
-

30. Can Apex support Multiple Inheritance?

- **Answer:** No, but can implement multiple interfaces.
-

31. What are Enums in Apex?

- **Answer:** Special data type representing constants.
 - **Example:**
 - `public enum Status { NEW, INPROGRESS, CLOSED }`
-

32. What are Transient variables?

- **Answer:** Variables not saved during serialization (mainly in Visualforce controllers).
-

33. What is the difference between `==` and `equals()` in Apex?

- **Answer:**
 - `==` checks for reference and primitive equality.
 - `.equals()` checks for logical equality.
-

34. What are Wrapper Classes?

- **Answer:** Custom classes that combine multiple data types/objects for complex logic.
-

35. Example of Wrapper Class?

- **Answer:**
 - `public class AccountWrapper {`
 - `public Account acc;`
 - `public Boolean isSelected;`
 - `}`
-

36. What is a Static Resource?

- **Answer:** Uploaded files (CSS, JS, images, ZIP) that can be used in VF, LWC, or Apex.
-

37. Can we perform DML in a Constructor?

- **Answer:** Yes, but not recommended as it can cause unexpected behavior.
-

38. What is the difference between Insert and Database.insert?

- **Answer:**
 - insert: Throws exception if any record fails.
 - Database.insert: Allows partial success using allOrNone=false.
-

39. What is the use of Schema class in Apex?

- **Answer:** Used for Dynamic Apex to access metadata at runtime.
 - **Example:** `Schema.sObjectType.Account.fields.getMap()`
-

40. What are Custom Metadata Types?

- **Answer:** Configuration data defined by admin/developer, deployable via packages, unlike custom settings.
-

41. What are Custom Settings?

- **Answer:** Similar to custom objects, but optimized for app configuration and accessible in Apex without SOQL.
-

42. What is the difference between Custom Metadata and Custom Settings?

- **Answer:**
 - **Custom Metadata:** Deployable, versioned, accessible in formulas.
 - **Custom Settings:** Not deployable via packages.
-

43. What are Annotations in Apex?

- **Answer:** Special keywords starting with @ that provide metadata to classes/methods.
 - **Examples:** @future, @isTest, @AuraEnabled.
-

44. What is @AuraEnabled used for?

- **Answer:** To expose Apex methods/properties to LWC/Aura components.
-

45. What is a Virtual Method?

- **Answer:** A method in a virtual class that can be overridden in child classes.
-

46. What is the use of instanceof operator?

- **Answer:** Checks if an object is an instance of a specific class.

- **Example:** `if(obj instanceof Account) { ... }`
-

47. What are Getter and Setter methods?

- **Answer:** Methods used to access and update private variables.
-

48. What is the difference between `get()` and `set()` methods in Apex properties?

- **Answer:**
 - **`get()`:** Retrieves value.
 - **`set()`:** Assigns value.
-

49. What is a Database Savepoint?

- **Answer:** Marks a point in a transaction to rollback if needed.
 - **Example:**
 - `Savepoint sp = Database.setSavepoint();`
 - `Database.rollback(sp);`
-

50. What is a Trigger Context Variable?

- **Answer:** Built-in variables like `Trigger.new`, `Trigger.old`, `Trigger.isInsert` that provide context during trigger execution.
-

◆ Part 2: Salesforce Triggers (40 Q&A)

1. What is a Trigger in Salesforce?

- **Answer:** A trigger is an Apex script that executes before or after DML operations on Salesforce records.
 - **Example:** Auto-populating a field when a record is created.
-

2. What are the two types of Triggers?

- **Answer:** Before triggers and After triggers.
 - **Explanation:**
 - **Before:** Used for validation/modifying values before saving.
 - **After:** Used for actions like creating related records.
-

3. What events can a Trigger run on?

- **Answer:** before insert, before update, before delete, after insert, after update, after delete, after undelete.
-

4. When should you use a Before Trigger?

- **Answer:** When you need to validate or update field values before they are committed.
 - **Example:** Setting BillingCity = ShippingCity.
-

5. When should you use an After Trigger?

- **Answer:** When records already exist in the database and you need record IDs or related records.
 - **Example:** Creating child records after parent insert.
-

6. What is Bulkification in Triggers?

- **Answer:** Writing code to handle multiple records in a single transaction instead of one-by-one.
 - **Example:** Using for (Account acc : Trigger.new) instead of single DML per record.
-

7. Why is Bulkification important?

- **Answer:** To avoid hitting governor limits and ensure performance in bulk operations (like Data Loader imports).
-

8. Give an example of a Bulkified Trigger.

```
trigger AccountTrigger on Account (before insert) {  
    for (Account acc : Trigger.new) {  
        if (acc.Industry == null) {  
            acc.Industry = 'Default';  
        }  
    }  
}
```

9. What are Trigger Context Variables?

- **Answer:** Built-in variables that provide context like `Trigger.new`, `Trigger.old`, `Trigger.isInsert`.
-

10. Difference between `Trigger.new` and `Trigger.old`?

- **Answer:**
 - **Trigger.new:** New record values in insert/update.
 - **Trigger.old:** Previous record values in update/delete.
-

11. Can we modify `Trigger.old`?

- **Answer:** No, it's read-only. Only `Trigger.new` (in before context) can be modified.
-

12. What is `Trigger.newMap` and `Trigger.oldMap`?

- **Answer:** Maps of record IDs to sObjects.

- **Example:** Trigger.newMap.get(acc.Id)
-

13. Difference between before insert and after insert?

- **Answer:**
 - **before insert:** Records not committed yet, can modify values.
 - **after insert:** Records saved, IDs available, can insert related records.
-

14. What is a Recursive Trigger?

- **Answer:** When a trigger's DML causes the same trigger to run again, creating an infinite loop.
-

15. How to prevent Recursive Triggers?

- **Answer:** Use a static Boolean variable or Trigger Handler framework.
 - **Example:**
 - `if(!MyHandler.hasRun) {`
 - `MyHandler.hasRun = true;`
 - `// logic`
 - `}`
-

16. What is the Trigger Order of Execution?

- **Answer:** A sequence Salesforce follows (system validation → before triggers → validation rules → after triggers → workflow → assignment rules → processes → commit).
-

17. What is a Trigger Handler Class?

- **Answer:** A separate Apex class that contains trigger logic, improving reusability and maintainability.

18. Why use Trigger Handler Pattern?

- **Answer:** To avoid hardcoding logic inside triggers, improve readability, and enforce best practices.

19. Example of Trigger Handler Pattern?

```
trigger AccountTrigger on Account (before insert) {  
    AccountTriggerHandler.handleBeforeInsert(Trigger.new);  
}
```

20. What is a Context-Specific Trigger?

- **Answer:** A trigger written to handle only one specific event.
- **Example:** A trigger that only runs for after delete.

21. Can we call a Future method inside a Trigger?

- **Answer:** Yes, but only from a non-batch context. Useful for callouts or async processing.

22. Can we perform Callouts from Triggers?

- **Answer:** Not directly. Must use @future or Queueable Apex.

23. Can we use DML inside a Trigger?

- **Answer:** Yes, but should be bulkified and optimized.

24. What happens if a Trigger has unhandled exceptions?

- **Answer:** The entire DML operation fails and no records are committed.
-

25. Can we write multiple Triggers on the same object?

- **Answer:** Yes, but not recommended. Execution order is unpredictable.
-

26. Best practice for multiple Triggers on same object?

- **Answer:** Have only one trigger per object, and delegate logic to handler classes.
-

27. Can we call a Batch class from a Trigger?

- **Answer:** Yes, but not recommended unless async processing is required.
-

28. What is an After Undelete Trigger?

- **Answer:** Trigger that fires when records are restored from Recycle Bin.
-

29. Example use case of After Undelete Trigger?

- **Answer:** Restoring child records or logs when parent record is undeleted.
-

30. What are Recursive DML Statements?

- **Answer:** When DML in a trigger causes the same trigger to run repeatedly.
-

31. Can you access Parent fields in a Trigger?

- **Answer:** Yes, using relationship queries.
 - **Example:** acc.Owner.Email if queried with SOQL.
-

32. What is the difference between Trigger.new and Database.insert?

- **Answer:** Trigger.new gives unsaved records in memory, while Database.insert performs actual DML.

33. Can we write Triggers on Standard Objects?

- **Answer:** Yes (e.g., Account, Contact, Opportunity).
-

34. Can we write Triggers on Custom Objects?

- **Answer:** Yes, works the same as standard objects.
-

35. Can we disable a Trigger temporarily?

- **Answer:** No direct toggle. Can use custom settings/metadata flags to conditionally bypass.
-

36. Can we run a Trigger only for a specific profile/user?

- **Answer:** Yes, by checking `UserInfo.getProfileId()` or custom logic.
-

37. Can a Trigger call another Trigger?

- **Answer:** Not directly, but recursive execution can cause re-invocation indirectly.
-

38. How do you handle Mixed DML Error in Triggers?

- **Answer:** Use `@future/Queueable` Apex when modifying setup and non-setup objects in same transaction.
-

39. Can you write Triggers on User object?

- **Answer:** Yes, but be cautious of Mixed DML errors.
-

40. What is the maximum number of records `Trigger.new` can hold?

- **Answer:** Up to **200 records** per trigger execution (due to batch size).

◆ Part 3: Asynchronous Apex (40 Q&A)

1. What is Asynchronous Apex?

- **Answer:** Apex code that runs in the background, outside the main transaction.
- **Explanation:** Used for long-running or resource-heavy tasks like callouts, batch processing, or large data updates.

2. Why do we use Asynchronous Apex?

- **Answer:** To avoid governor limits in synchronous transactions and improve performance.
- **Example:** Sending emails or making callouts after DML.

3. What are the types of Asynchronous Apex?

- **Answer:** Future Methods, Batch Apex, Queueable Apex, Schedulable Apex.

4. What is a Future Method?

- **Answer:** A method annotated with @future that runs asynchronously.
- **Example:** Sending callouts or email notifications after DML.

5. Syntax of Future Method?

@future

```
public static void updateContact(String name) {  
    // logic  
}
```

6. Limitations of Future Methods?

- **Answer:**
 - Cannot return values.
 - Max 50 calls per transaction.
 - Cannot chain future methods.
-

7. What is Queueable Apex?

- **Answer:** Similar to future methods but supports complex data types and chaining.
 - **Example:** `System.enqueueJob(new MyQueueable());`
-

8. Syntax of Queueable Apex?

```
public class MyQueueable implements Queueable {  
    public void execute(QueueableContext context) {  
        // logic  
    }  
}
```

9. Benefits of Queueable Apex over Future?

- **Answer:**
 - Can chain jobs.
 - Can pass complex objects.
 - Provides job ID for monitoring.
-

10. Can we call a Future method inside a Queueable class?

- **Answer:** No, chaining Future methods is not allowed.

11. What is Batch Apex?

- **Answer:** Apex designed to process large volumes of data (millions of records) asynchronously in chunks.
- **Example:** Recalculating fields for all Accounts.

12. Syntax of Batch Apex?

```
global class MyBatch implements Database.Batchable<sObject> {  
    global Database.QueryLocator start(Database.BatchableContext bc) {  
        return Database.getQueryLocator('SELECT Id FROM Account');  
    }  
    global void execute(Database.BatchableContext bc, List<Account> scope) {  
        // process  
    }  
    global void finish(Database.BatchableContext bc) {  
        // post-processing  
    }  
}
```

13. How to run a Batch?

- **Answer:** Database.executeBatch(new MyBatch(), 200);

14. What is the default batch size?

- **Answer:** 200 records per execution.

15. What is the maximum batch size?

- **Answer:** 2000 records per execution.

16. Can we call Future methods in Batch?

- **Answer:** No, but you can call Queueable Apex instead.
-

17. What is the maximum number of Batch jobs queued?

- **Answer:** 5 active/queued batch jobs at a time.
-

18. What is Database.Stateful in Batch Apex?

- **Answer:** Interface that maintains variable state across batch executions.
 - **Example:** Storing totals across batch chunks.
-

19. Can we do Callouts in Batch Apex?

- **Answer:** Yes, by implementing Database.AllowsCallouts.
-

20. Can Batch Apex be scheduled?

- **Answer:** Yes, via Schedulable Apex or System.scheduleBatch.
-

21. What is Schedulable Apex?

- **Answer:** Apex that runs at a specific time/interval using CRON expressions.
 - **Example:** Running nightly data cleanup jobs.
-

22. Syntax of Schedulable Apex?

```
global class MyScheduler implements Schedulable {  
    global void execute(SchedulableContext sc) {  
        System.debug('Scheduled job running');  
    }  
}
```

}

23. How to schedule an Apex job?

- **Answer:**
- String cron = '0 0 23 * * ?';
- System.schedule('DailyJob', cron, new MyScheduler());

24. What is a CRON expression in Salesforce?

- **Answer:** String that defines schedule timing (seconds, minutes, hours, day, month, weekday, optional year).

25. Can we schedule a job from UI?

- **Answer:** Yes, via **Setup → Apex Classes → Schedule Apex**.

26. What is the limit for scheduled jobs?

- **Answer:** Max **100 scheduled jobs** per org.

27. Can we chain Queueable jobs?

- **Answer:** Yes, only one additional job per execution.

28. Difference between Queueable and Batch Apex?

- **Answer:**
 - **Queueable:** Small async tasks, chainable.
 - **Batch:** Large datasets, divided into chunks.

29. Can Queueable Apex be scheduled?

- **Answer:** Indirectly, by calling inside a Schedulable class.

30. What is a One-Off Batch Execution?

- **Answer:** Running a batch once using Database.executeBatch.
-

31. Can Batch Apex run indefinitely?

- **Answer:** No, limited to 5 days execution.
-

32. What is the maximum execution time for a Queueable job?

- **Answer:** Up to 60 seconds CPU time.
-

33. Can we use Database.Savepoint in Async Apex?

- **Answer:** No, savepoints and rollbacks are not supported in async jobs.
-

34. Can we monitor Async jobs?

- **Answer:** Yes, via **Setup → Apex Jobs** or AsyncApexJob object.
-

35. What is AsyncApexJob object?

- **Answer:** Standard object that stores info about async jobs (status, completion, errors).
-

36. How to handle errors in Batch Apex?

- **Answer:** Use try-catch inside execute() and log failed records separately.
-

37. Can we chain Batch jobs?

- **Answer:** Yes, by calling another batch inside finish() method.
-

38. Difference between Future and Batch Apex?

- **Answer:**
 - **Future:** Lightweight, simple async tasks.
 - **Batch:** Heavy, large data processing in chunks.
-

39. Difference between Batch and Scheduled Apex?

- **Answer:**
 - **Batch:** Processes large data.
 - **Scheduled:** Runs jobs at specific times (can trigger batch inside).
-

40. When would you choose Queueable Apex over Batch?

- **Answer:** For **medium-volume tasks** that need chaining or complex parameter passing, but don't require chunking.
-

◆ Part 4: Lightning Web Components (60 Q&A)

1. What is LWC?

- **Answer:** Lightning Web Components is a modern framework built on web standards (ES6, HTML, CSS) for building UI in Salesforce.
-

2. Why was LWC introduced?

- **Answer:** To replace Aura with a faster, standards-based framework, improve performance, and reduce learning curve.
-

3. Difference between LWC and Aura?

- **Answer:**
 - LWC is built on web standards.
 - Aura is proprietary Salesforce framework.
 - LWC has better performance and less boilerplate code.

4. What are Decorators in LWC?

- **Answer:** Special annotations that modify component behavior: @api, @track, @wire.
-

5. What is @api in LWC?

- **Answer:** Makes a property/method public so it can be passed from parent to child component.
-

6. What is @track in LWC?

- **Answer:** Used for reactivity of private fields in older LWC versions (mostly replaced by reactive variables).
-

7. What is @wire in LWC?

- **Answer:** Used to read Salesforce data declaratively or call Apex methods.
-

8. Example of @wire with Apex?

```
@wire(getAccounts) accounts;
```

9. What is the difference between @wire and imperative Apex calls?

- **Answer:**
 - **@wire:** Reactive, auto-refresh, declarative.
 - **Imperative:** Called manually with JS function.
-

10. Example of Imperative Apex call?

```
import getAccounts from '@salesforce/apex/AccountController.getAccounts';  
getAccounts().then(result => {...}).catch(error => {...});
```

11. What is the component lifecycle in LWC?

- **Answer:** constructor → connectedCallback → renderedCallback → disconnectedCallback → errorCallback.
-

12. What is connectedCallback?

- **Answer:** Lifecycle hook called when component is inserted into DOM.
-

13. What is renderedCallback?

- **Answer:** Lifecycle hook called after every render of the component.
-

14. What is disconnectedCallback?

- **Answer:** Called when component is removed from DOM.
-

15. What is errorCallback?

- **Answer:** Called when child component throws an error.
-

16. What are Lightning Data Service (LDS) adapters in LWC?

- **Answer:** Prebuilt adapters like getRecord, getRecordUi, updateRecord for CRUD without Apex.
-

17. Example of using getRecord?

@wire(getRecord, { recordId: '\$recordId', fields: [NAME_FIELD] }) account;

18. What is Lightning Message Service (LMS)?

- **Answer:** Pub-sub framework for communication between components (LWC, Aura, VF).

19. Example of publishing a message in LMS?

```
publish(this.messageContext, SAMPLEMC, { data: 'Hello' });
```

20. Example of subscribing in LMS?

```
subscribe(this.messageContext, SAMPLEMC, (message) => { this.msg =  
message.data; });
```

21. How do you handle Parent to Child communication in LWC?

- **Answer:** Pass values via @api decorated properties.
-

22. How do you handle Child to Parent communication?

- **Answer:** Use CustomEvent and dispatch it from child.
-

23. Example of dispatching CustomEvent?

```
this.dispatchEvent(new CustomEvent('myevent', { detail: value }));
```

24. What are Slots in LWC?

- **Answer:** Placeholders inside a component where parent markup can be injected.
-

25. Example of Slot usage?

```
<slot></slot>
```

26. What are Lightning Base Components?

- **Answer:** Prebuilt UI components like lightning-input, lightning-card, lightning-datatable.

27. What is lightning-datatable?

- **Answer:** A standard component to display tabular data with sorting, inline editing, pagination.
-

28. How to handle form input in LWC?

- **Answer:** Use lightning-input and capture value in JS with event.target.value.
-

29. How to perform Navigation in LWC?

- **Answer:** Use NavigationMixin from lightning/navigation.
-

30. Example of navigation to record page?

```
this[NavigationMixin.Navigate]({  
  type: 'standard__recordPage',  
  attributes: { recordId: this.recordId, objectApiName: 'Account', actionName:  
    'view' }  
});
```

31. What is wire service caching?

- **Answer:** Salesforce automatically caches wire results for performance.
-

32. How to refresh a wire adapter?

- **Answer:** Use refreshApex(result).
-

33. Can LWC call REST API directly?

- **Answer:** Yes, using fetch() with Named Credentials/remote site settings.

34. How to import Static Resource in LWC?

import myImage from '@salesforce/resourceUrl/myImage';

35. How to import Custom Label in LWC?

import LABEL_NAME from '@salesforce/label/c.LabelName';

36. How to import Custom Permission in LWC?

import hasPermission from '@salesforce/customPermission/My_Permission';

37. How to import User Info in LWC?

import Id from '@salesforce/user/Id';

38. How do you handle conditional rendering in LWC?

- **Answer:** Use <template if:true={property}> and <template if:false={property}>.
-

39. What is a Reactive Property in LWC?

- **Answer:** A property that triggers re-render when updated.
-

40. What is two-way data binding in LWC?

- **Answer:** LWC doesn't support full two-way binding, only one-way binding + event handling.
-

41. What is Lightning Locker?

- **Answer:** A security framework that restricts DOM access, prevents cross-component scripting.

42. What is Lightning Web Security (LWS)?

- **Answer:** Successor to Locker, provides stricter sandboxing but better compatibility.
-

43. Can LWC be used outside Salesforce?

- **Answer:** Yes, via **Lightning Out** or **Open Source LWC**.
-

44. What is LWC OSS?

- **Answer:** Open Source version of LWC that runs outside Salesforce.
-

45. What are Wire Adapters?

- **Answer:** Prebuilt @wire services like `getRecord`, `getObjectInfo`.
-

46. What is `getObjectInfo`?

- **Answer:** Wire adapter to fetch object metadata like label, record type.
-

47. Example of `getObjectInfo`?

```
@wire(getObjectInfo, { objectApiName: ACCOUNT_OBJECT }) objectInfo;
```

48. How do you access field labels dynamically?

- **Answer:** Using `getFieldValue(record.data, FIELD_NAME)` from `lightning/uiRecordApi`.
-

49. What is LDS form in LWC?

- **Answer:** `lightning-record-form`, `lightning-record-view-form`, `lightning-record-edit-form`.

50. Difference between lightning-record-form and record-edit-form?

- **Answer:**
 - record-form: Auto generates UI.
 - record-edit-form: More control over fields.
-

51. How to optimize LWC performance?

- **Answer:** Use caching, reduce DOM re-renders, use imperative calls only when needed.
-

52. What is Shadow DOM in LWC?

- **Answer:** Encapsulation mechanism for styles and DOM inside a component.
-

53. How do you apply CSS in LWC?

- **Answer:** Use component-specific CSS file (componentName.css).
-

54. Can we use CSS from static resource?

- **Answer:** Yes, by importing and applying dynamically.
-

55. What is composition in LWC?

- **Answer:** Building complex UI by nesting components.
-

56. What is Lightning App Builder in LWC context?

- **Answer:** Allows admins to drag-drop LWCs onto pages.
-

57. How do you make a component available in App Builder?

- **Answer:** Define targets in meta.xml file.

<targets>

<target>lightning__RecordPage</target>

</targets>

58. Can LWC call Apex methods without cache?

- **Answer:** Yes, using **imperative calls**.
-

59. What are limitations of LWC?

- **Answer:**
 - Cannot directly access DOM outside component.
 - No two-way binding.
 - Limited server-side rendering.
-

60. When to use LWC vs Aura?

- **Answer:** Use LWC for new development; Aura only when using features not yet supported in LWC.
-

◆ Part 5: Aura Components & Migration to LWC (20 Q&A)

1. What is Aura?

- **Answer:** Aura is Salesforce's legacy component framework used before LWC, based on proprietary programming model.
-

2. Difference between Aura and LWC?

- **Answer:**
 - Aura → Proprietary, slower, more boilerplate.

- LWC → Web standards, faster, better performance.

3. What is an Aura Component bundle?

- **Answer:** A set of files including .cmp, .controller.js, .helper.js, .css, .design, .svg, .auradoc.

4. What is Application Event in Aura?

- **Answer:** Used for communication between unrelated components (like pub-sub model).

5. What is Component Event in Aura?

- **Answer:** Used for communication between parent and child components.

6. What are the phases of Event propagation in Aura?

- **Answer:** Capture → Bubble.

7. What is Aura Handler?

- **Answer:** <aura:handler> tag used to listen to component or application events.

8. Example of Aura component event?

<aura:event type="COMPONENT" name="cmpEvent"/>

9. What are Attributes in Aura?

- **Answer:** Properties defined in <aura:attribute> used to store values in component.
-

10. Example of defining Aura attribute?

```
<aura:attribute name="accName" type="String" default="Test"/>
```

11. How does Data Binding work in Aura?

- **Answer:** Uses expression syntax `{!v.attribute}` for one-way and `{#v.attribute}` for two-way binding.
-

12. What is difference between Aura Controller and Helper?

- **Answer:**
 - **Controller.js** → Handles UI events.
 - **Helper.js** → Contains reusable functions for business logic.
-

13. How to call Apex from Aura?

- **Answer:** Use `@AuraEnabled` in Apex, then call via `component.get("c.methodName")`.
-

14. Example of calling Apex in Aura?

```
var action = component.get("c.getAccounts");  
action.setCallback(this, function(response) {...});  
$A.enqueueAction(action);
```

15. Why migrate from Aura to LWC?

- **Answer:** Better performance, easier maintenance, uses standard JS, better future support.
-

16. Can Aura and LWC work together?

- **Answer:** Yes, LWC can be embedded inside Aura using `<c:myLwcComp />`.
-

17. Can Aura listen to events from LWC?

- **Answer:** Yes, LWC CustomEvent can be caught in Aura with `<aura:handler>`.
-

18. Can LWC use Aura Events?

- **Answer:** No, LWC cannot directly use Aura events (must use LMS or custom communication).
-

19. Migration challenges from Aura to LWC?

- **Answer:**
 - Rewriting events (Aura → CustomEvent / LMS).
 - CSS isolation differences.
 - Some missing APIs in LWC.
-

20. Best practices for migrating Aura to LWC?

- **Answer:**
 - Start with simple reusable components.
 - Replace component events with CustomEvent.
 - Replace application events with LMS.
 - Use LWC for new features, Aura for legacy until fully migrated.
-
-

◆ Part 6: Visualforce (15 Q&A)

1. What is Visualforce?

- **Answer:** A framework for building custom user interfaces in Salesforce using HTML-like markup and server-side Apex controllers.
-

2. What is a Visualforce Page?

- **Answer:** A page created using <apex:page> tag, rendered on Salesforce UI, often backed by an Apex controller.
-

3. What is the difference between Standard Controller and Custom Controller?

- **Answer:**
 - **Standard Controller:** Uses Salesforce standard objects.
 - **Custom Controller:** Apex class with full custom logic.
-

4. What is an Extension in Visualforce?

- **Answer:** An Apex class that extends functionality of a standard controller.
-

5. Example of a Visualforce page with Standard Controller?

```
<apex:page standardController="Account">
  <apex:form>
    <apex:inputField value="{!Account.Name}"/>
    <apex:commandButton action="{!save}" value="Save"/>
  </apex:form>
</apex:page>
```

6. What is the difference between Controller and Extension?

- **Answer:**
 - **Controller:** Primary class linked to VF page.

- **Extension:** Adds extra logic to Standard Controller without overriding it.
-

7. How do you pass parameters in Visualforce?

- **Answer:** Using <apex:param> or URL parameters. Example: /apex/MyPage?id=001xx000003DGbY.
-

8. What is the difference between apex:pageBlock and apex:pageBlockTable?

- **Answer:**
 - pageBlock: Creates a styled section.
 - pageBlockTable: Displays data in a table format.
-

9. How do you call Apex methods from Visualforce?

- **Answer:** Using action methods in controller (public PageReference myMethod()) and referencing in action="{!myMethod}".
-

10. What is apex:actionFunction?

- **Answer:** Defines a JavaScript function that can call an Apex controller method.
-

11. What is apex:actionPoller?

- **Answer:** Periodically calls a controller method at a set interval.
-

12. How can you include a Lightning Component in Visualforce?

- **Answer:** Using lightning:out or ltng:require.
-

13. Can Visualforce and LWC communicate?

- **Answer:** Yes, via Lightning Message Service (LMS) or Lightning Out.
-

14. What are Visualforce Remoting methods?

- **Answer:** JavaScript functions that call Apex methods annotated with @RemoteAction.
-

15. When should you use Visualforce vs LWC?

- **Answer:**
 - **Visualforce:** For classic pages, legacy applications.
 - **LWC:** For modern Lightning Experience and new development.
-

◆ Part 7: SOQL & SOSL (30 Q&A)

1. What is SOQL?

- **Answer:** Salesforce Object Query Language, used to retrieve records from Salesforce objects.
-

2. What is SOSL?

- **Answer:** Salesforce Object Search Language, used to search text across multiple objects simultaneously.
-

3. Difference between SOQL and SOSL?

- **Answer:**
 - **SOQL:** Query structured data from one object at a time (with subqueries).
 - **SOSL:** Search text, phone, email across multiple objects.
-

4. Can SOQL query multiple objects?

- **Answer:** Yes, via **relationship queries** (parent-to-child subqueries or child-to-parent joins).
-

5. Example of SOQL Parent-to-Child query?

SELECT Name, (SELECT LastName FROM Contacts) FROM Account

6. Example of SOQL Child-to-Parent query?

SELECT LastName, Account.Name FROM Contact

7. Example of SOSL?

FIND 'Acme*' IN ALL FIELDS RETURNING Account(Name), Contact(LastName, Email)

8. Can SOSL return multiple objects in a single query?

- **Answer:** Yes, unlike SOQL which returns from one root object at a time.
-

9. What is the maximum number of SOQL queries per transaction?

- **Answer:** 100 (synchronous) and 200 (asynchronous).
-

10. What is the maximum number of records SOQL can return?

- **Answer:** 50,000 records.
-

11. What is the governor limit for SOSL queries?

- **Answer:** 20 SOSL queries per transaction.
-

12. What are Aggregate Functions in SOQL?

- **Answer:** Functions like COUNT(), SUM(), AVG(), MIN(), MAX().
-

13. Example of COUNT in SOQL?

SELECT COUNT() FROM Account

14. Example of GROUP BY in SOQL?

SELECT Industry, COUNT(Id) FROM Account GROUP BY Industry

15. What is HAVING in SOQL?

- **Answer:** Used with aggregate functions to filter grouped results.

SELECT Industry, COUNT(Id) FROM Account GROUP BY Industry HAVING COUNT(Id) > 10

16. Can we use ORDER BY in SOQL?

- **Answer:** Yes, to sort results ASC or DESC.
-

17. What is LIMIT in SOQL?

- **Answer:** Restricts number of records returned.

SELECT Name FROM Account LIMIT 10

18. What is OFFSET in SOQL?

- **Answer:** Skips a given number of rows, useful for pagination.
-

19. Difference between Semi-Join and Anti-Join?

- **Answer:**
 - **Semi-Join:** Finds records with related child records.
 - **Anti-Join:** Finds records without related child records.

20. Example of Semi-Join?

SELECT Id, Name FROM Account WHERE Id IN (SELECT AccountId FROM Contact)

21. Example of Anti-Join?

SELECT Id, Name FROM Account WHERE Id NOT IN (SELECT AccountId FROM Contact)

22. What is FOR UPDATE in SOQL?

- **Answer:** Locks selected records so other users can't modify them until transaction completes.
-

23. Can you use LIKE in SOQL?

- **Answer:** Yes, for partial string matching.

SELECT Name FROM Account WHERE Name LIKE 'Acme%'

24. Can SOQL query formula fields?

- **Answer:** Yes, but values are calculated at runtime.
-

25. Can SOQL query encrypted fields?

- **Answer:** Yes, but only equality filters are supported, not LIKE or ORDER BY.
-

26. What is relationship query limit in SOQL?

- **Answer:** Up to **35 child-to-parent** relationships and **1 parent-to-child** subquery per query.
-

27. Can SOQL return records from Recycle Bin?

- **Answer:** Yes, using ALL ROWS keyword.

SELECT Name FROM Account WHERE IsDeleted = true ALL ROWS

28. Can SOQL query fields from multiple objects without a relationship?

- **Answer:** No, unless using SOSL or a junction object.
-

29. How to optimize SOQL queries?

- **Answer:** Use **selective filters, indexed fields, avoid SELECT, use LIMIT, use skinny tables.*
-

30. When to use SOQL vs SOSL?

- **Answer:**
 - **SOQL:** Structured queries when you know the object.
 - **SOSL:** Search when you don't know where data exists (like global search).
-
-

◆ Part 8: Integration (40 Q&A)

1. What are the different types of Salesforce Integration?

- **Answer:**
 - API-based (REST, SOAP)
 - Event-based (Platform Events, Change Data Capture, Streaming API)
 - Middleware-based (Mulesoft, Dell Boomi, Informatica)
-

2. What is REST API in Salesforce?

- **Answer:** A lightweight API using HTTP methods (GET, POST, PATCH, DELETE) to interact with Salesforce data in JSON format.
-

3. What is SOAP API?

- **Answer:** A protocol-based API that uses XML for structured communication.
-

4. Difference between REST API and SOAP API?

- **Answer:**
 - REST → Lightweight, JSON, stateless.
 - SOAP → Heavy, XML, strict contract (WSDL).
-

5. What is Named Credential?

- **Answer:** A secure way to store authentication details for external API calls without hardcoding credentials in Apex.
-

6. Example of making REST callout in Apex?

```
HttpRequest req = new HttpRequest();  
req.setEndpoint('callout:MyAPI');  
req.setMethod('GET');  
Http http = new Http();  
HttpResponse res = http.send(req);
```

7. What is Remote Site Setting?

- **Answer:** A Salesforce configuration that allows callouts to external endpoints (mandatory if not using Named Credentials).
-

8. What are Callouts in Salesforce?

- **Answer:** Requests made from Salesforce to external systems via REST/SOAP.
-

9. Can callouts be made from Triggers?

- **Answer:** No, callouts are not allowed directly. Must use `@future(callout=true)` or **Queueable Apex**.
-

10. What is Continuation in Salesforce?

- **Answer:** Used for **long-running callouts** (async) to avoid hitting limits like 120-second timeout.
-

11. What is Bulk API?

- **Answer:** An API optimized for loading/unloading large data volumes (asynchronous, batch-based).
-

12. What is Streaming API?

- **Answer:** Push-based integration that notifies clients in real-time when Salesforce data changes.
-

13. What is Platform Event?

- **Answer:** A Salesforce entity used for **event-driven architecture** to publish/subscribe messages.
-

14. What is Change Data Capture (CDC)?

- **Answer:** Event-based mechanism that publishes changes (create, update, delete, undelete) to records in real time.
-

15. Difference between Platform Events and CDC?

- **Answer:**
 - Platform Events: Custom events, defined by developer.
 - CDC: Automatically captures record changes in standard/custom objects.
-

16. What is Outbound Messaging?

- **Answer:** A declarative integration tool in Workflow/Flow that sends SOAP message to external systems.
-

17. What is Apex Web Service?

- **Answer:** An Apex class exposed as a SOAP or REST service using @WebService or @RestResource.
-

18. Example of exposing Apex as REST service?

@RestResource(urlMapping='/myApi/*')

global with sharing class MyService {

 @HttpGet

 global static String doGet() {

 return 'Hello World';

 }

}

19. Example of exposing Apex as SOAP service?

global with sharing class MySoapService {

 WebService static String getMsg(String name) {

 return 'Hello ' + name;

```
}  
  
}
```

20. How do you test callouts in Apex tests?

- **Answer:** Use `HttpCalloutMock` to simulate responses, since actual callouts are not allowed in tests.

21. What is Middleware in Salesforce integration?

- **Answer:** A software layer (e.g., MuleSoft, Dell Boomi) used to connect Salesforce with other systems, handling transformations, retries, orchestration.

22. What is OData Connector in Salesforce?

- **Answer:** Allows Salesforce to access external data via **External Objects** using OData protocol.

23. What are External Objects?

- **Answer:** Read-only representations of data stored outside Salesforce, accessed via **Salesforce Connect**.

24. What is the difference between Salesforce Connect and Data Loader?

- **Answer:**
 - Salesforce Connect → Real-time access (no data storage).
 - Data Loader → Imports data into Salesforce.

25. How do you secure integrations in Salesforce?

- **Answer:** Using OAuth 2.0, Named Credentials, IP restrictions, Connected Apps.

26. What is Connected App in Salesforce?

- **Answer:** A framework to allow external applications to authenticate with Salesforce using OAuth.
-

27. What is OAuth 2.0 in Salesforce?

- **Answer:** An authentication protocol that allows secure delegated access to Salesforce APIs.
-

28. Different OAuth flows supported by Salesforce?

- **Answer:**
 - Authorization Code Flow
 - Username-Password Flow
 - JWT Bearer Flow
 - Client Credentials Flow
-

29. What is difference between Authorization Code Flow and JWT Flow?

- **Answer:**
 - Auth Code Flow → Interactive login, user presence needed.
 - JWT Flow → Server-to-server, no user interaction.
-

30. What is EventBus in Salesforce?

- **Answer:** The mechanism that stores and delivers events (Platform Events, CDC) within Salesforce.
-

31. How long are Platform Events retained in Salesforce?

- **Answer:** 72 hours.

32. What is ReplayId in Streaming API/Platform Events?

- **Answer:** A unique ID assigned to each event, used to retrieve missed events.
-

33. What is Named Credential vs Custom Metadata for Integration?

- **Answer:**
 - Named Credential → Authentication + Endpoint.
 - Custom Metadata → Stores configs like API keys but not auth.
-

34. What is Inbound vs Outbound Integration in Salesforce?

- **Answer:**
 - Inbound → External system calls Salesforce API.
 - Outbound → Salesforce makes callout to external system.
-

35. What is the daily API request limit in Salesforce?

- **Answer:** Depends on edition & license (Enterprise: 100,000 per day minimum).
-

36. What is the difference between REST Composite API and Batch API?

- **Answer:**
 - Composite API → Combine multiple REST calls in one request.
 - Batch API → Execute multiple independent requests in parallel.
-

37. What is Pub/Sub API?

- **Answer:** A Salesforce API for event-driven integrations (supports high throughput, gRPC protocol).

38. What are common integration patterns in Salesforce?

- **Answer:**
 - Remote Call-In (Inbound)
 - Remote Process Invocation (Outbound)
 - Batch Data Sync
 - Event-driven messaging
-

39. When would you use Middleware vs Direct API?

- **Answer:**
 - Middleware → Complex transformations, orchestration, multiple systems.
 - Direct API → Simple point-to-point integration.
-

40. What are best practices for Salesforce integration?

- **Answer:**
 - Use Named Credentials for authentication.
 - Avoid hardcoding endpoints/keys.
 - Use Platform Events for async integrations.
 - Respect governor limits.
 - Implement retry mechanisms & error handling.
-
-

◆ Part 9: OmniStudio (30 Q&A)

1. What is OmniStudio in Salesforce?

- **Answer:** A digital engagement suite that allows developers and admins to build guided interactions, integrations, and UI without heavy coding.

2. What are the main components of OmniStudio?

- **Answer:** OmniScripts, DataRaptors, Integration Procedures, FlexCards, and Expression Sets.
-

3. What is an OmniScript?

- **Answer:** A guided flow that walks users step-by-step through business processes (like wizards).
-

4. What are the advantages of OmniScripts?

- **Answer:** Declarative, reusable, configurable, reduces need for custom LWC/Apex.
-

5. What is a DataRaptor?

- **Answer:** A mapping tool to extract, transform, and load (ETL) Salesforce data.
-

6. Types of DataRaptors?

- **Answer:**
 - **Extract** → Fetch data from Salesforce.
 - **Load** → Insert/Update Salesforce.
 - **Transform** → Format/manipulate data.
 - **Turbo Extract** → High-performance data extract.
-

7. What is an Integration Procedure?

- **Answer:** Server-side process to perform multiple actions (DataRaptors, Apex, HTTP callouts) in one call.
-

8. Why use Integration Procedures?

- **Answer:** To optimize performance by reducing multiple client-server calls into one.
-

9. What is a FlexCard?

- **Answer:** A UI component that displays Salesforce/external data in a customizable card layout.
-

10. Use cases of FlexCards?

- **Answer:** Display account summary, case history, policy details, billing information.
-

11. What is the difference between FlexCards and OmniScripts?

- **Answer:**
 - FlexCard → Display data.
 - OmniScript → Guided process for user input.
-

12. Can FlexCards be embedded in LWC?

- **Answer:** Yes, FlexCards can be deployed inside LWCs, Communities, or as standalone.
-

13. What is Expression Set in OmniStudio?

- **Answer:** A configuration to evaluate conditions and return true/false, often used for branching logic.
-

14. How do you pass data between OmniScript and Integration Procedure?

- **Answer:** Using JSON data structures.
-

15. How do you optimize DataRaptor performance?

- **Answer:** Use Turbo Extract, filter data at query level, minimize fields, avoid nested structures.
-

16. What are OmniStudio Actions?

- **Answer:** Prebuilt functionalities like navigate, save, post message, call integration.
-

17. How does versioning work in OmniStudio?

- **Answer:** Each OmniScript, FlexCard, or DataRaptor can have multiple versions, ensuring safe changes without affecting production.
-

18. What is OmniStudio Designer?

- **Answer:** A UI-based tool in Salesforce where OmniScripts, DataRaptors, and FlexCards are built.
-

19. What are DataRaptor Interface Limits?

- **Answer:** Max 5 DataRaptors per Integration Procedure for best performance.
-

20. How do you migrate OmniStudio components between orgs?

- **Answer:** Using **IDX Workbench** or Salesforce packages.
-

21. What is IDX Workbench?

- **Answer:** A migration tool for OmniStudio that allows moving assets between environments.
-

22. What is OmniStudio Console?

- **Answer:** A UI container to host OmniScripts, FlexCards, and provide agent workspaces.
-

23. How do you debug OmniScripts?

- **Answer:** Use **Preview mode** and **Tracking** logs to analyze step execution.
-

24. How do you handle external API calls in OmniStudio?

- **Answer:** Use **Integration Procedures** with HTTP Action elements.
-

25. Can OmniScripts be embedded in Experience Cloud?

- **Answer:** Yes, they can be exposed to portals and communities.
-

26. What is a common use case for OmniScripts?

- **Answer:** Guided KYC process, insurance claims, telecom onboarding, loan application.
-

27. What are best practices for building OmniScripts?

- **Answer:** Modular design, reusable actions, minimize steps, use Integration Procedures for heavy logic.
-

28. How does error handling work in OmniStudio?

- **Answer:** Integration Procedures support error blocks, and OmniScripts can display custom error messages.
-

29. What is the benefit of combining FlexCards with OmniScripts?

- **Answer:** FlexCard shows data; OmniScript allows user action on that data in one interface.
-

30. Difference between OmniStudio vs Flow?

- **Answer:**
 - Flow: Native Salesforce automation tool.
 - OmniStudio: More advanced, optimized for industry-specific guided flows & integrations.
-
-

◆ Part 10: Sales Cloud (25 Q&A)

1. What is Sales Cloud in Salesforce?

- **Answer:** A CRM solution to manage leads, opportunities, accounts, contacts, forecasting, and sales processes.
-

2. What are the key objects in Sales Cloud?

- **Answer:** Lead, Account, Contact, Opportunity, Campaign, Quote, Forecast.
-

3. What is a Lead in Sales Cloud?

- **Answer:** A prospect who has shown interest but is not yet qualified.
-

4. What is Lead Conversion?

- **Answer:** The process of converting a lead into Account, Contact, and optionally Opportunity.
-

5. Can a lead be converted without creating an Opportunity?

- **Answer:** Yes, Salesforce provides the option to skip creating an Opportunity during conversion.
-

6. What is an Opportunity?

- **Answer:** A potential revenue-generating deal linked to an Account and Contact.
-

7. What are Opportunity Stages?

- **Answer:** Different phases of the sales cycle (e.g., Prospecting, Negotiation, Closed Won, Closed Lost).
-

8. What is Opportunity Probability?

- **Answer:** A percentage value that reflects the likelihood of closing a deal, linked to the Opportunity Stage.
-

9. What is Forecasting in Sales Cloud?

- **Answer:** A tool to predict future revenue based on Opportunities in the pipeline.
-

10. What is Collaborative Forecasting?

- **Answer:** A forecasting tool that allows managers and reps to adjust and share forecasts in real time.
-

11. Difference between Campaign and Opportunity?

- **Answer:**
 - Campaign → Marketing initiative to generate leads.
 - Opportunity → Sales deal that generates revenue.
-

12. How are Campaigns linked to Opportunities?

- **Answer:** Through Campaign Influence, which tracks the impact of campaigns on Opportunities.
-

13. What are Quotes in Sales Cloud?

- **Answer:** Price proposals for Opportunities, containing product, pricing, and discount details.
-

14. Can multiple Quotes be linked to one Opportunity?

- **Answer:** Yes, but only one Quote can be marked as the **primary quote**.
-

15. What is Price Book in Sales Cloud?

- **Answer:** A collection of products and their prices that can be linked to Opportunities or Quotes.
-

16. What is a Product in Sales Cloud?

- **Answer:** A catalog item or service that can be sold through Opportunities and Quotes.
-

17. What is an Order in Sales Cloud?

- **Answer:** A contract to provide products or services, typically created after a Quote is accepted.
-

18. What is Territory Management?

- **Answer:** A feature to assign accounts and opportunities to sales reps based on geographic or business rules.
-

19. What is Contact Role on Opportunity?

- **Answer:** Identifies the role of a Contact (e.g., Decision Maker, Influencer) in an Opportunity.
-

20. What is a Path in Sales Cloud?

- **Answer:** A guided process UI that helps users progress through Opportunity or Lead stages.
-

21. How do you automate Lead Assignment?

- **Answer:** Using Lead Assignment Rules based on criteria like region, product, or source.
-

22. What is a Lead Queue?

- **Answer:** A holding area where unassigned leads are stored until a rep claims them.
-

23. What is a Big Deal Alert?

- **Answer:** A notification system that alerts sales teams about large Opportunities.
-

24. What is Einstein Opportunity Scoring?

- **Answer:** An AI feature that predicts the likelihood of closing an Opportunity based on historical data.
-

25. Best practices for customizing Sales Cloud?

- **Answer:**
 - Automate lead routing & follow-ups.
 - Use validation rules to ensure data quality.
 - Implement Path and Guidance for Success.
 - Use Forecasting for pipeline visibility.
-

◆ Part 11: Service Cloud (25 Q&A)

1. What is Service Cloud in Salesforce?

- **Answer:** A Salesforce solution designed for customer service and support, providing tools for case management, omni-channel routing, and knowledge management.
-

2. What is a Case in Service Cloud?

- **Answer:** A record that tracks customer issues, questions, or requests.
-

3. What are Case Assignment Rules?

- **Answer:** Rules that automatically assign cases to users or queues based on predefined criteria.
-

4. What is a Case Queue?

- **Answer:** A holding area where unassigned cases are stored until an agent takes ownership.
-

5. What is Omni-Channel in Service Cloud?

- **Answer:** A feature that intelligently routes work items (cases, chats, tasks) to the right agent based on skills, capacity, and availability.
-

6. Difference between Omni-Channel and Case Assignment Rules?

- **Answer:**
 - **Case Assignment Rules** → Assign cases to users/queues at creation.
 - **Omni-Channel** → Routes real-time work dynamically based on agent workload.
-

7. What are Entitlements in Service Cloud?

- **Answer:** Define the level of support customers are entitled to, such as response times and resolution times.

8. What is a Milestone in Service Cloud?

- **Answer:** A measurable step within an entitlement process, e.g., "First Response within 2 hours."
-

9. What is a Service Level Agreement (SLA) in Salesforce?

- **Answer:** An agreement defining the time frame within which a case must be responded to or resolved, implemented using entitlements and milestones.
-

10. What is a Knowledge Base in Service Cloud?

- **Answer:** A repository of articles and FAQs that help agents and customers resolve issues quickly.
-

11. Difference between Salesforce Knowledge and Solution?

- **Answer:**
 - **Solution (Legacy)** → Simple Q&A for cases.
 - **Knowledge** → Advanced knowledge base with publishing workflows, versioning, and multi-channel access.
-

12. What is a Service Console?

- **Answer:** A unified workspace for agents that allows handling multiple cases, chats, and customer interactions in one view.
-

13. What is a Case Escalation Rule?

- **Answer:** Rules that automatically escalate cases to higher support levels if they remain unresolved within a certain time.
-

14. What are Macros in Service Cloud?

- **Answer:** Automated actions that let agents perform repetitive tasks (like sending standard responses) in one click.
-

15. What is Live Agent (Chat) in Salesforce?

- **Answer:** A real-time chat feature that connects customers with support agents through websites or apps.
-

16. What is Einstein Bots in Service Cloud?

- **Answer:** AI-powered chatbots that handle routine customer queries before handing them off to live agents.
-

17. What is Case Feed?

- **Answer:** A Chatter-based collaboration feature that allows agents to communicate and update case details in real-time.
-

18. What is Email-to-Case?

- **Answer:** A feature that automatically converts customer emails into cases.
-

19. What is Web-to-Case?

- **Answer:** A feature that allows customers to submit cases directly from a website form.
-

20. What is Social Customer Service?

- **Answer:** A Service Cloud feature that lets companies respond to customer inquiries from social media (Twitter, Facebook, etc.).
-

21. What are Service Contracts in Salesforce?

- **Answer:** Agreements that define support terms and cover customers, often linked with entitlements.
-

22. What is Field Service Lightning (FSL)?

- **Answer:** A Salesforce product that extends Service Cloud for managing mobile workforce, scheduling, and field operations.
-

23. How does Knowledge Base integrate with Cases?

- **Answer:** Agents can search and attach knowledge articles directly to case records for quick resolutions.
-

24. What is the use of Service Analytics?

- **Answer:** Provides dashboards and reports on KPIs like case resolution time, first contact resolution, agent productivity.
-

25. Best practices in Service Cloud implementation?

- **Answer:**
 - Automate case routing and escalation.
 - Use macros for efficiency.
 - Enable Omni-Channel for workload balancing.
 - Leverage Knowledge Base for faster resolutions.
-

◆ Part 12: Experience Cloud (25 Q&A)

1. What is Experience Cloud in Salesforce?

- **Answer:** A Salesforce product that allows organizations to create branded portals, partner communities, and customer self-service sites.

2. What are the different types of Experience Cloud sites?

- **Answer:** Customer communities, partner communities, employee communities, B2B portals, and self-service help portals.
-

3. Difference between Customer Community and Partner Community?

- **Answer:**
 - **Customer Community** → For end customers to log cases, access knowledge, and collaborate.
 - **Partner Community** → For business partners to manage leads, opportunities, and deals.
-

4. What is the license difference between Customer and Partner Community?

- **Answer:**
 - Customer Community License → Focused on service/support use cases.
 - Partner Community License → Includes CRM objects like Leads and Opportunities.
-

5. What is the difference between Experience Cloud and Sites (Force.com Sites)?

- **Answer:**
 - Experience Cloud → Branded, customizable, authenticated or public.
 - Force.com Sites → Public websites only, no login functionality.
-

6. How is Experience Cloud secured?

- **Answer:** Security is managed using profiles, permission sets, sharing sets, and role hierarchies.

7. What are Sharing Sets in Experience Cloud?

- **Answer:** Provide community users access to records related to their account or contact.
-

8. What is a Role Hierarchy in Partner Communities?

- **Answer:** Defines how partner users can see and manage records within their partner organization.
-

9. What is a Login & Registration page in Experience Cloud?

- **Answer:** The entry point where external users authenticate or sign up to access the portal.
-

10. What authentication methods are available for Experience Cloud?

- **Answer:** Salesforce login, Single Sign-On (SSO), Social Sign-On (Google, Facebook, LinkedIn).
-

11. What is Delegated Authentication?

- **Answer:** Allows an external system to validate user credentials instead of Salesforce.
-

12. What is Just-in-Time (JIT) provisioning in Experience Cloud?

- **Answer:** Creates user records automatically during the first SSO login.
-

13. What is Identity Provider (IdP) vs Service Provider (SP) in Experience Cloud?

- **Answer:**
 - IdP → System that authenticates users (e.g., Okta, Salesforce).

- SP → System that users access (Experience Cloud site).

14. What is Branding in Experience Cloud?

- **Answer:** Customizing the look and feel (logo, theme, colors, navigation) of a portal using Experience Builder.

15. What is the Experience Builder?

- **Answer:** A drag-and-drop tool to design and customize Experience Cloud sites without coding.

16. Can LWC be used in Experience Cloud?

- **Answer:** Yes, LWCs can be exposed as Lightning components and embedded into Experience Cloud pages.

17. What are Audience Targeting rules in Experience Cloud?

- **Answer:** Allow different page variations to be shown based on user profiles, location, or criteria.

18. What is a Navigation Menu in Experience Cloud?

- **Answer:** Defines how users move between pages and features within the portal.

19. How are Knowledge Articles exposed in Experience Cloud?

- **Answer:** By enabling Knowledge in the site and giving access to external users via permissions.

20. What are the types of Community users?

- **Answer:** Customer Community User, Partner Community User, Customer Plus User, External App User.
-

21. What are Community Moderation features?

- **Answer:** Flags, reputation points, and moderation rules to monitor and control user activity.
-

22. What are Reputation Levels in Experience Cloud?

- **Answer:** A gamification feature that rewards users for participation (posting, commenting, answering).
-

23. What is the use of CMS Connect in Experience Cloud?

- **Answer:** Integrates external CMS content (like WordPress, Drupal) into Experience Cloud.
-

24. How do you deploy an Experience Cloud site to production?

- **Answer:** By publishing the site in Experience Builder and deploying metadata using Change Sets or SFDX.
-

25. Best practices for Experience Cloud implementation?

- **Answer:**
 - Use sharing sets for simplified record access.
 - Optimize performance with caching.
 - Apply audience targeting for personalization.
 - Ensure strong authentication and branding.
-

◆ Part 13: Security & Sharing

(30 Q&A)

1. What are the different levels of data security in Salesforce?

- **Answer:**
 1. **Object-level** → Profiles/Permission Sets.
 2. **Field-level** → Field-Level Security (FLS).
 3. **Record-level** → OWD, Roles, Sharing Rules, Manual Sharing.
-

2. What is Organization-Wide Default (OWD)?

- **Answer:** Defines the baseline level of access to records for all users in the org (Private, Public Read Only, Public Read/Write, Controlled by Parent).
-

3. What is the difference between Profile and Permission Set?

- **Answer:**
 - Profile → Defines baseline permissions (mandatory for a user).
 - Permission Set → Grants additional access without changing profile.
-

4. Can a user exist without a Profile in Salesforce?

- **Answer:** No, every user must be assigned exactly one Profile.
-

5. Can a user have multiple Permission Sets?

- **Answer:** Yes, a user can have multiple permission sets.
-

6. What is a Role in Salesforce?

- **Answer:** Defines a user's position in the role hierarchy and determines record visibility via role-based sharing.

7. Difference between Role Hierarchy and Sharing Rules?

- **Answer:**
 - Role Hierarchy → Opens access vertically (upward visibility).
 - Sharing Rules → Opens access horizontally (across peers).
-

8. What is Manual Sharing?

- **Answer:** A record owner or admin can manually share a record with another user.
-

9. What is Apex Managed Sharing?

- **Answer:** Programmatic sharing of records using Apex to insert into Share tables.
-

10. What are Share Tables?

- **Answer:** Special tables (e.g., AccountShare, OpportunityShare) that store record-level sharing information.
-

11. Can we share records via SOQL/DML directly?

- **Answer:** No, but we can insert/update records in Share objects for programmatic sharing.
-

12. What is Criteria-Based Sharing Rule?

- **Answer:** Grants record access automatically based on field values (e.g., share all Cases where Priority = High with a group).
-

13. What is the difference between “With Sharing” and “Without Sharing” in Apex?

- **Answer:**

- With Sharing → Enforces the sharing rules of the user.
 - Without Sharing → Runs in system mode, ignoring sharing rules.
-

14. What is System Mode in Salesforce?

- **Answer:** Execution mode where Apex code ignores user permissions and sharing rules.
-

15. Difference between Sharing Rules and Permission Sets?

- **Answer:**
 - Sharing Rules → Control **record-level** access.
 - Permission Sets → Control **object/field-level** access.
-

16. Can we restrict access using Sharing Rules?

- **Answer:** No, sharing rules only grant additional access; restrictions must be done using OWD.
-

17. What is a Public Group in Salesforce?

- **Answer:** A collection of users, roles, and subgroups used to simplify sharing rules.
-

18. What is a Queue in Salesforce?

- **Answer:** A shared location where records (like Cases, Leads) are assigned until a user takes ownership.
-

19. Difference between Public Group and Queue?

- **Answer:**

- Public Group → Used for record sharing.
 - Queue → Used for record ownership assignment.
-

20. What is Implicit Sharing?

- **Answer:** Automatic record sharing by Salesforce, such as parent-child record access (e.g., Contact to Account).
-

21. What is Enterprise Territory Management (ETM)?

- **Answer:** A feature to control access to Accounts and Opportunities based on territories like geography or product line.
-

22. What is “Grant Access Using Hierarchies”?

- **Answer:** A setting that ensures higher roles in hierarchy automatically get access to lower-level records.
-

23. Can “Grant Access Using Hierarchies” be disabled?

- **Answer:**
 - For **standard objects** → Always enforced (cannot disable).
 - For **custom objects** → Can be disabled.
-

24. What is Profile vs Role difference?

- **Answer:**
 - Profile → Defines **what** a user can do (permissions).
 - Role → Defines **what records** a user can see (visibility).
-

25. What is Field-Level Security (FLS)?

- **Answer:** Controls visibility (read/edit) of specific fields at the profile or permission set level.
-

26. What is Record Ownership?

- **Answer:** The user who creates a record is typically its owner and controls who else can see/edit it.
-

27. What is Row-Level Security?

- **Answer:** Another term for record-level access, managed through OWD, sharing rules, and manual sharing.
-

28. Can we use Apex to bypass FLS and CRUD?

- **Answer:** Yes, Apex runs in system mode, so explicit checks (`Schema.sObjectType.Account.fields.Name.isAccessible()`) are needed.
-

29. What is a Sharing Set in Experience Cloud?

- **Answer:** A feature that grants community users access to records associated with their account/contact.
-

30. Best practices for Security & Sharing in Salesforce?

- **Answer:**
 - Follow **principle of least privilege**.
 - Use Permission Sets instead of cloning profiles.
 - Prefer OWD + Sharing Rules over Apex sharing.
 - Check FLS and CRUD in Apex for compliance.
-

1. What are the different deployment methods in Salesforce?

- **Answer:** Change Sets, Ant Migration Tool, Salesforce DX (SFDX), Unmanaged/Managed Packages, Third-party DevOps tools (Gearset, Copado, Flosum).
-

2. What is a Change Set?

- **Answer:** A point-and-click deployment tool that allows metadata to be moved between connected Salesforce orgs (e.g., Sandbox → Production).
-

3. Limitations of Change Sets?

- **Answer:**
 - Only between related orgs.
 - No rollback option.
 - Manual dependency management.
 - Not supported for all metadata types.
-

4. What is Ant Migration Tool?

- **Answer:** A Java-based command-line tool that uses the Metadata API to deploy metadata between Salesforce orgs.
-

5. What is Salesforce DX (SFDX)?

- **Answer:** A modern CLI-based tool for source-driven development, version control, scratch orgs, and CI/CD automation.
-

6. What is a Scratch Org in SFDX?

- **Answer:** A temporary, source-driven Salesforce org used for development and testing, usually with a lifespan of 1–30 days.

7. Difference between Sandbox and Scratch Org?

- **Answer:**
 - Sandbox → Persistent copy of production used for testing.
 - Scratch Org → Disposable, customizable org for development.
-

8. What are the types of Sandboxes in Salesforce?

- **Answer:** Developer, Developer Pro, Partial Copy, Full Copy.
-

9. Difference between Developer Sandbox and Full Sandbox?

- **Answer:**
 - Developer → Limited storage, no production data.
 - Full → Complete copy of production with all data.
-

10. What is Continuous Integration (CI) in Salesforce?

- **Answer:** A practice of automatically validating, testing, and merging code changes using tools like GitHub Actions, Jenkins, or GitLab CI.
-

11. What is Continuous Deployment (CD)?

- **Answer:** Automating the process of pushing validated code from version control into Salesforce environments.
-

12. What is the difference between Continuous Delivery vs Deployment?

- **Answer:**
 - Continuous Delivery → Automated testing, but deployment may still require approval.

- Continuous Deployment → Fully automated push to production with no manual intervention.
-

13. What are Managed vs Unmanaged Packages?

- **Answer:**
 - Managed → For AppExchange, version-controlled, locked.
 - Unmanaged → For distribution, open and editable.
-

14. What are unlocked packages?

- **Answer:** A packaging option in SFDX that allows modular deployments with versioning for enterprise teams.
-

15. What is Git in Salesforce DevOps?

- **Answer:** A version control system used to track changes to Salesforce metadata and coordinate among developers.
-

16. What is the difference between Metadata API and Tooling API?

- **Answer:**
 - Metadata API → Deploy and retrieve metadata.
 - Tooling API → Manage developer-specific metadata like Apex classes, triggers, debug logs.
-

17. What is Rollback in Deployment?

- **Answer:** Undoing a failed deployment; Salesforce doesn't support native rollback, but metadata can be re-deployed from version control.
-

18. What are Pre-Deployment and Post-Deployment steps?

- **Answer:**

- Pre → Disable triggers, validation rules, workflows.
 - Post → Re-enable them and run smoke tests.
-

19. What is a Deployment Validation?

- **Answer:** Running a deployment simulation with all tests without committing changes to production.
-

20. What is a Deployment Connection?

- **Answer:** The trust relationship between Salesforce orgs that allows Change Set deployments.
-

21. What is a Deployment Plan?

- **Answer:** A step-by-step strategy outlining components, dependencies, pre/post tasks, rollback plan, and testing strategy.
-

22. What is Static Code Analysis in Salesforce?

- **Answer:** Tools like PMD, SonarQube, CodeScan used to enforce best practices before deployment.
-

23. What is Code Coverage requirement for deployment?

- **Answer:** 75% overall test coverage is required to deploy Apex to production.
-

24. What is Test Level in Deployment?

- **Answer:** Options: Run Local Tests, Run All Tests, Run Specified Tests. Used to control which test classes execute during deployment.
-

25. Difference between "Run Local Tests" and "Run All Tests"?

- **Answer:**

- Local Tests → Runs only org-specific test classes.
 - All Tests → Runs managed package tests too.
-

26. What is Test Isolation in Salesforce?

- **Answer:** Test classes run independently with their own data context; test data is not shared across tests.
-

27. What is Metadata API format vs SFDX format?

- **Answer:**
 - Metadata API → Folder-based format for Change Sets and Ant.
 - SFDX Format → Source-driven, modular folder structure for version control.
-

28. What is the role of a DevOps Center in Salesforce?

- **Answer:** Salesforce's own DevOps tool for managing changes with pipelines, Git integration, and deployment tracking.
-

29. What are common DevOps tools used in Salesforce?

- **Answer:** Copado, Gearset, AutoRABIT, Flosun, Jenkins, GitHub Actions.
-

30. Best practices for Salesforce Deployment?

- **Answer:**
 - Always deploy from version control.
 - Use SFDX and unlocked packages.
 - Validate deployments before release.
 - Automate CI/CD pipelines.

- Maintain rollback strategy.
-

◆ Part 15: Governor Limits & Performance Tuning (25 Q&A)

1. What are Governor Limits in Salesforce?

- **Answer:** Salesforce-imposed runtime limits to ensure efficient use of resources in a multi-tenant environment.
-

2. Why do we have Governor Limits?

- **Answer:** To prevent one org's code from monopolizing shared Salesforce resources.
-

3. Name some common Apex Governor Limits.

- **Answer:**
 - SOQL queries: 100 per transaction.
 - DML statements: 150 per transaction.
 - Records retrieved by SOQL: 50,000.
 - Heap size: 6 MB (synchronous), 12 MB (async).
 - Callouts: 100 per transaction.
-

4. What is the limit on CPU time?

- **Answer:** 10,000 ms for synchronous and 60,000 ms for asynchronous transactions.
-

5. What is the limit on future calls?

- **Answer:** 50 future calls per transaction.
-

6. What are Limits methods in Salesforce?

- **Answer:** The Limits class provides methods like getDmlStatements(), getQueries(), getHeapSize(), etc. to check consumption.
-

7. What happens when governor limits are exceeded?

- **Answer:** The system throws a **runtime exception** and the transaction is rolled back.
-

8. How to handle “Too many SOQL queries: 101” error?

- **Answer:** Bulkify code → Use collections, maps, and avoid queries inside loops.
-

9. How to handle “Too many DML statements: 151” error?

- **Answer:** Bulkify DML by grouping records into lists and using a single insert/update call.
-

10. What is Heap Size limit?

- **Answer:** Maximum memory allocated to store objects in memory during execution (6 MB sync, 12 MB async).
-

11. How to reduce Heap Size errors?

- **Answer:**
 - Use transient keyword in Visualforce controllers.
 - Stream large data instead of storing in memory.
 - Query only required fields.
-

12. What is the maximum batch size in Batch Apex?

- **Answer:** 2,000 records per batch.

13. How to handle “Too many queueable jobs added” error?

- **Answer:** Max 50 Queueable jobs per transaction; chain jobs carefully.
-

14. What is the maximum callout limit per transaction?

- **Answer:** 100 HTTP callouts.
-

15. What is the maximum size of a SOQL query result?

- **Answer:** 50,000 records.
-

16. What is the maximum size of SOSL query results?

- **Answer:** 2,000 records.
-

17. How do you avoid hitting CPU time limits?

- **Answer:**
 - Optimize loops.
 - Use maps and sets.
 - Offload heavy logic to asynchronous Apex.
-

18. How to tune queries for performance?

- **Answer:**
 - Use **selective queries** with indexed fields.
 - Use LIMIT keyword.
 - Avoid null checks in filters.
-

19. What is a Selective Query in Salesforce?

- **Answer:** A query that uses indexed fields in WHERE clauses and retrieves <10% of object records.
-

20. How to check if a query is selective?

- **Answer:** Use **Query Plan Tool** in Developer Console.
-

21. What is View State in Visualforce?

- **Answer:** The serialized data sent between server and client to maintain page state; max size 135 KB.
-

22. How to reduce View State size?

- **Answer:**
 - Use transient variables.
 - Use smaller forms.
 - Avoid large collections in controllers.
-

23. What is the best practice for handling large data volumes (LDV)?

- **Answer:**
 - Use indexed fields.
 - Use skinny tables.
 - Use asynchronous processing (Batch, Queueable, Platform Events).
-

24. What is a Skinny Table?

- **Answer:** Custom database tables created by Salesforce support to improve query performance on large data sets.
-

25. Best practices to avoid hitting Governor Limits?

- **Answer:**

- Always bulkify triggers and Apex.
 - Use collections for queries/DML.
 - Use batch jobs for large data.
 - Monitor with Limits class.
 - Prefer declarative automation when possible.
-

◆ Part 16: Testing (25 Q&A)

1. Why do we need testing in Salesforce?

- **Answer:** To ensure code works as expected, prevent regressions, meet 75% code coverage requirement, and validate business logic.
-

2. What is the minimum test coverage required for production deployment?

- **Answer:** 75% across all Apex classes and triggers, and each trigger must have some coverage.
-

3. What is @isTest annotation?

- **Answer:** Marks a class or method as a test class/method. Such methods don't count against org limits and aren't included in package size.
-

4. What is seeAllData=true in test classes?

- **Answer:** Allows test methods to access existing org data. Best practice: **avoid using it** and create test data instead.
-

5. How to create test data in Apex tests?

- **Answer:** By inserting test records in the test method or using a `@testSetup` method.
-

6. What is the `@testSetup` annotation?

- **Answer:** Creates common test data once per class, shared across all test methods for efficiency.
-

7. Difference between `test.startTest()` and `test.stopTest()`?

- **Answer:**
 - `startTest()` resets governor limits.
 - `stopTest()` executes asynchronous code (future, batch, queueable, schedulable).
-

8. Can we call DML in test classes?

- **Answer:** Yes, but the data is rolled back automatically after the test completes.
-

9. Can we commit data permanently from a test class?

- **Answer:** No, data inserted in tests is not committed to the database.
-

10. Can we test private methods in Apex?

- **Answer:** Yes, using `@TestVisible` annotation.
-

11. How do we test triggers?

- **Answer:** By creating test data that executes trigger conditions and verifying expected results using `System.assert`.
-

12. How to test a future method?

- **Answer:** Call the method inside a test and wrap in `Test.startTest()` and `Test.stopTest()`.
-

13. How to test a batch class?

- **Answer:** Use `Database.executeBatch(new BatchClass(), 200)` inside test methods.
-

14. How to test a Queueable class?

- **Answer:** Use `System.enqueueJob(new QueueableClass())` inside a test method.
-

15. How to test a Schedulable class?

- **Answer:** Use `System.schedule('Test Job', '0 0 0 ? * * *', new MySchedulableClass())` inside a test.
-

16. What is a Test Utility Class?

- **Answer:** A helper class with static methods to create reusable test data, improving maintainability.
-

17. How to test callouts in Salesforce?

- **Answer:** Use `HttpCalloutMock` or `WebServiceMock` interfaces to simulate responses.
-

18. What is HttpCalloutMock?

- **Answer:** A test interface to define mock HTTP responses for test methods that perform callouts.
-

19. How to test Platform Events?

- **Answer:** Publish events in test classes and subscribe using triggers or processes to validate logic.
-

20. How to test Exception Scenarios?

- **Answer:** Use System.assertException or try-catch blocks in tests to verify proper handling.
-

21. Can we exclude classes from code coverage?

- **Answer:** Yes, by marking them with @isTest or @TestVisible where applicable.
-

22. What is the maximum test execution time limit?

- **Answer:** 60 minutes per test method.
-

23. How to check code coverage in Salesforce?

- **Answer:** Developer Console, Setup → Apex Test Execution, or sfdx force:apex:test:run.
-

24. What are best practices for writing test classes?

- **Answer:**
 - Cover positive & negative cases.
 - Avoid seeAllData=true.
 - Use @testSetup for reusability.
 - Cover bulk scenarios.
 - Use assertions to validate results.
-

25. What is the difference between unit test and integration test in Salesforce?

- **Answer:**

- Unit Test → Tests a single class or method in isolation.
 - Integration Test → Tests interaction between multiple components (triggers, classes, flows, etc.).
-

◆ Part 17: Design Patterns & Frameworks (20 Q&A)

1. What are design patterns in Salesforce?

- **Answer:** Standardized coding solutions to common problems (e.g., Trigger Handler, Singleton, Factory, Strategy). They improve maintainability and scalability.
-

2. What is the Trigger Framework pattern?

- **Answer:** A structure that organizes trigger logic by delegating operations to a handler class, preventing multiple triggers per object.
-

3. Why is a Trigger Framework important?

- **Answer:**
 - Ensures **one trigger per object**.
 - Improves readability and maintainability.
 - Supports bulkification and recursion handling.
-

4. Can you explain the Trigger Handler pattern?

- **Answer:** A class with methods like `beforeInsert`, `beforeUpdate`, `afterInsert`, etc., that contains trigger logic separated from the trigger itself.
-

5. How do you prevent recursion in triggers?

- **Answer:**

- Use a static Boolean variable.
 - Use a trigger framework with recursion control.
-

6. What is the Singleton design pattern?

- **Answer:** Ensures only one instance of a class is created and shared. Useful for managing shared resources like configuration data.
-

7. How is Singleton implemented in Apex?

- **Answer:** By using a **private constructor** and a **static method** that returns a single instance.
-

8. What is the Factory design pattern?

- **Answer:** A pattern that creates objects without exposing the creation logic. Useful for instantiating classes dynamically.
-

9. Example use case of Factory pattern in Salesforce?

- **Answer:** Creating different payment gateway integrations (PayPal, Stripe, Razorpay) based on configuration.
-

10. What is the Strategy design pattern?

- **Answer:** Defines a family of algorithms and lets the client choose one at runtime. Promotes flexibility.
-

11. Example use case of Strategy pattern in Salesforce?

- **Answer:** Different discount calculation strategies (student discount, seasonal discount, bulk discount) applied dynamically.
-

12. What is the Command design pattern?

- **Answer:** Encapsulates a request as an object, allowing execution, queuing, or logging of commands. Often used in batch or async processing.
-

13. What is the Observer design pattern?

- **Answer:** A pattern where one object (subject) notifies multiple dependent objects (observers) when its state changes.
 - Example: Platform Events notifying multiple subscribers.
-

14. What is the Decorator design pattern?

- **Answer:** Adds new functionality to an existing object dynamically without modifying its structure.
 - Example: Adding logging or auditing logic.
-

15. What is the Adapter design pattern?

- **Answer:** Converts the interface of one class into another expected by clients.
 - Example: Wrapping an external REST API response into Salesforce standard format.
-

16. What is the Repository design pattern?

- **Answer:** Provides a centralized location for data access logic, separating business logic from SOQL queries.
 - Example: A repository class handling all Account queries.
-

17. What is the Unit of Work pattern?

- **Answer:** Groups multiple DML operations into a single transaction, reducing governor limit usage. Available in **Apex Enterprise Patterns**.

18. What is Dependency Injection in Apex?

- **Answer:** Passing dependencies (objects or services) into a class rather than hardcoding them. Promotes testability and flexibility.
-

19. What are Apex Enterprise Patterns?

- **Answer:** A set of design patterns (by Andrew Fawcett) that includes **Service, Domain, Selector, and Unit of Work layers** for scalable Apex applications.
-

20. What are best practices for using design patterns in Salesforce?

- **Answer:**
 - Use trigger frameworks.
 - Apply Singleton for config caching.
 - Use Factory for dynamic instantiation.
 - Apply Unit of Work for bulk DML.
 - Avoid over-engineering; use patterns only when they add value.
-

◆ Part 18: Advanced Topics (30 Q&A)

1. What are Custom Settings in Salesforce?

- **Answer:** Custom objects used to store configuration data accessible in Apex without SOQL queries. Types: List and Hierarchy.
-

2. Difference between Custom Settings and Custom Metadata?

- **Answer:**
 - **Custom Settings** → Data is org-specific, not deployable with metadata.

- **Custom Metadata** → Deployable via metadata, version-controlled, better for app configurations.
-

3. When to use Custom Metadata over Custom Settings?

- **Answer:** Use **Custom Metadata** when configuration needs to be deployed across orgs (e.g., feature flags, API endpoints).
-

4. What is Dynamic Apex?

- **Answer:** Apex features that allow runtime access to objects, fields, and SOQL queries using Schema and Describe methods.
-

5. Example use case of Dynamic Apex?

- **Answer:** Building a generic data import utility that can read field definitions dynamically.
-

6. What is Describe Information in Apex?

- **Answer:** Metadata about objects/fields (label, datatype, picklist values) retrieved using Schema.DescribeSObjectResult.
-

7. What is the difference between Dynamic SOQL and Static SOQL?

- **Answer:**
 - **Static SOQL** → Written at compile time (SELECT Id, Name FROM Account).
 - **Dynamic SOQL** → Built at runtime using strings (Database.query(queryString)).
-

8. What is Tooling API in Salesforce?

- **Answer:** API that allows developers to build custom tools for metadata (Apex classes, triggers, Visualforce, logs) management.

9. Difference between Metadata API and Tooling API?

- **Answer:**
 - **Metadata API** → Deploys/retrieves metadata.
 - **Tooling API** → Used for developer tools (debug logs, code coverage, Apex compilation).
-

10. What are Custom Labels in Salesforce?

- **Answer:** Text values stored in Salesforce for multi-language support, accessed in Apex, LWC, and Visualforce.
-

11. What is the purpose of Platform Events?

- **Answer:** Enable event-driven architecture by allowing publishers and subscribers to communicate asynchronously.
-

12. Difference between Platform Events and Change Data Capture (CDC)?

- **Answer:**
 - **Platform Events** → Custom events defined by developers.
 - **CDC** → Out-of-the-box events generated when records change.
-

13. What is a Big Object?

- **Answer:** Special object type for storing and querying massive datasets (billions of records) in Salesforce.
-

14. How do you query Big Objects?

- **Answer:** Using SOQL with **Async SOQL** or **standard SOQL** (with limitations).
-

15. What are External Objects?

- **Answer:** Objects that map to data stored outside Salesforce using Salesforce Connect.
-

16. Difference between External Objects and Big Objects?

- **Answer:**
 - **External Objects** → Data stored outside Salesforce (real-time access).
 - **Big Objects** → Data stored inside Salesforce for historical/archival purposes.
-

17. What is a Named Credential?

- **Answer:** A secure way to store authentication details for external callouts, avoiding hardcoded credentials.
-

18. What is a Custom Metadata Loader?

- **Answer:** A utility (often an Apex + CSV solution) to bulk insert/update custom metadata records.
-

19. What is Apex Managed Sharing?

- **Answer:** Programmatically sharing records by creating entries in Share objects, used when declarative sharing is insufficient.
-

20. Difference between with sharing, without sharing, and inherited sharing?

- **Answer:**
 - **with sharing** → Respects user's sharing rules.
 - **without sharing** → Runs in system mode.
 - **inherited sharing** → Inherits caller's context.

21. What is the purpose of Custom Metadata in CI/CD?

- **Answer:** Helps in deploying environment-specific configurations (like endpoint URLs) without changing code.

22. What are Limits of Custom Metadata?

- **Answer:** Max 200KB per record, no encrypted fields, not suitable for frequently changing data.

23. What is Apex Dynamic Binding?

- **Answer:** Ability to refer to fields dynamically in SOQL/DML without hardcoding field names.

24. What is the difference between SOQL Injection and Dynamic SOQL?

- **Answer:**
 - **SOQL Injection** → Security risk when user input is directly concatenated in queries.
 - **Dynamic SOQL** → Safe if using bind variables or `escapeSingleQuotes`.

25. How to prevent SOQL injection in Apex?

- **Answer:**
 - Use **bind variables**.
 - Use `String.escapeSingleQuotes()`.
 - Validate user input.

26. What is Asynchronous SOQL?

- **Answer:** A feature to process large data volumes in parallel using async jobs (returns results via callback).
-

27. What is the Streaming API?

- **Answer:** Enables clients to receive push notifications when data changes in Salesforce.
-

28. What is the difference between Streaming API and Platform Events?

- **Answer:**
 - **Streaming API** → Predefined events based on PushTopics or CDC.
 - **Platform Events** → Developer-defined custom events.
-

29. What is the Tooling API used for in CI/CD?

- **Answer:** Fetching code coverage, running tests, retrieving debug logs, and analyzing deployed components.
-

30. Best practices for Advanced Apex development?

- **Answer:**
 - Use **Custom Metadata** for configs.
 - Avoid hardcoding IDs and URLs.
 - Use **Dynamic Apex** only when needed.
 - Use **Named Credentials** for callouts.
 - Use **Event-driven architecture** for scalability.
-

◆ Part 19: Real-time Scenarios / Use Cases (100+ Q&A)

1. How do you ensure a trigger is bulkified?

- Use collections (lists/maps/sets), avoid SOQL/DML inside loops.
-

2. Trigger needs to prevent duplicate Account creation based on Email. How would you implement it?

- Query existing Accounts with same Email → If found, use `addError()` on the record.
-

3. How do you avoid recursion in triggers?

- Use a **static Boolean flag** or recursion handler class.
-

4. A trigger is firing twice when inserting records via Data Loader. Why?

- Multiple triggers exist or workflow/process is updating same object → consolidate logic into one trigger framework.
-

5. You need to auto-populate BillingCity from ShippingCity. Where do you write this?

- In a **before insert, before update** trigger.
-

6. How do you handle trigger order of execution?

- Ensure logic respects Salesforce's standard order: validation → before triggers → DML → after triggers → workflows.
-

7. Can we call a future method from a trigger?

- Yes, but ensure limits (max 50) and avoid chaining unnecessarily.
-

8. How do you update parent records when child records are inserted?

- Use **after insert trigger** on child, aggregate values, then update parent.
-

9. Trigger needs to restrict Account deletion if it has active Opportunities. Solution?

- Query related Opportunities → If active found, addError() on Account.
-

10. Trigger must create a Contact automatically when Account is created. Where do you write this?

- **After insert trigger** on Account → Insert Contact records.
-

11. How do you write trigger logic reusable across objects?

- Use a **generic trigger handler framework**.
-

12. You're hitting "Too many SOQL queries: 101" in a trigger. Fix?

- Move queries outside loops, aggregate using IN operator.
-

13. Trigger needs to assign sequential numbers to Cases. How would you do it?

- Use a **Custom Setting/Metadata for last number** and update it in trigger.
-

14. How do you handle large data updates in triggers?

- Use **Batch Apex** instead of heavy trigger logic.
-

15. You need to send email notifications after a record update in trigger. Where should you do it?

- In **after update trigger**.
-

16. How do you handle field history tracking limits in triggers?

- Create a **custom audit object** and insert logs.

17. Trigger needs to handle file attachments migration. How?

- Use ContentDocumentLink object in after insert/update triggers.

18. What happens if you have multiple triggers on the same object?

- Execution order is not guaranteed → Best practice: one trigger per object.

19. How do you test trigger logic with bulk records?

- Insert 200+ test records in test class.

20. When would you choose Flow over Trigger?

- Use Flow for **declarative simple logic**, Trigger for **complex, scalable logic**.

 **B. Integration Scenarios (20 Q&A)**

21. How do you make a REST API callout from Salesforce?

- Use Http, HttpRequest, and HttpResponse classes.

22. How do you authenticate external API calls securely?

- Use **Named Credentials** instead of hardcoding credentials.

23. You need to schedule nightly sync with external system. Which Apex do you use?

- **Schedulable Apex + Batch Apex.**

24. How do you expose Salesforce data to external systems?

- Use **Apex REST services** or **SOAP web services**.

25. What is the difference between REST and SOAP in Salesforce?

- REST → Lightweight, JSON/XML. SOAP → Heavier, strict WSDL, enterprise use.
-

26. How do you handle large data integration?

- Use **Bulk API** or **Batch Apex** with callouts.
-

27. How do you retry failed callouts in Salesforce?

- Use **Queueable Apex** with retry logic.
-

28. How do you handle governor limits in API callouts?

- Move callouts to **Queueable/Future Apex**.
-

29. How to handle authentication tokens in integrations?

- Store tokens in **Custom Metadata/Named Credentials** and refresh programmatically.
-

30. How do you integrate Salesforce with external database?

- Use **Salesforce Connect** (External Objects) or middleware (MuleSoft).
-

31. Which API to use for real-time sync?

- **Streaming API / Platform Events**.
-

32. Which API to use for batch sync of millions of records?

- **Bulk API 2.0**.
-

33. How do you prevent duplicate API requests in Salesforce?

- Implement **idempotency key** in integration.
-

34. What is the limit of callouts per transaction?

- 100 callouts.
-

35. How do you debug API callout failures?

- Check **Debug Logs** and setup **Named Credential logs**.
-

36. How do you send data to external system when a record changes in Salesforce?

- Use **Outbound Messages**, **Platform Events**, or **Apex callouts**.
-

37. When do you use Middleware (MuleSoft, Dell Boomi, etc.) over direct API?

- When integration needs **transformation, orchestration, error handling**.
-

38. How do you handle governor limits when syncing thousands of records to an API?

- Process in **Batch Apex** with callouts per batch.
-

39. What's the difference between Named Credential and Remote Site Settings?

- Named Credential → Stores URL + Auth securely.
 - Remote Site Settings → Only whitelists URLs.
-

40. How do you expose Salesforce REST API to mobile apps?

- Use **Connected Apps + OAuth 2.0** authentication.
-

41. How do you pass data from parent to child LWC?

- Use **@api public property**.
-

42. How do you pass data from child to parent LWC?

- Fire **Custom Events**.
-

43. How do you call Apex from LWC?

- Use **@AuraEnabled** methods with **@wire** or imperative calls.
-

44. Which is better: wire vs imperative Apex call?

- **Wire** → Reactive, automatic refresh.
 - **Imperative** → Manual control.
-

45. How do you handle large data in LWC datatable?

- Use **pagination + lazy loading**.
-

46. How do you display related records dynamically in LWC?

- Use **getRelatedRecords** Apex method or SOQL.
-

47. How do you refresh data after record update in LWC?

- Use **refreshApex()**.
-

48. How do you navigate from LWC to record page?

- Use **NavigationMixin**.
-

49. Can LWC call external API directly?

- No, must go through **Apex callout** or **Named Credentials**.
-

50. How do you optimize LWC performance?

- Lazy load components, cache data (@wire with cacheable=true), reduce DOM re-renders.
-

51. How do you handle Lightning Message Service in LWC?

- Use lightning/messageService to communicate between LWCs or Aura components across DOM boundaries.
-

52. How do you prevent excessive Apex calls in LWC?

- Cache data locally, use @wire(cacheable=true) and combine multiple records in one call.
-

53. How do you handle form validation in LWC?

- Use reportValidity() and checkValidity() methods on lightning-input fields.
-

54. How do you handle error messages from Apex in LWC?

- Capture the error property returned by @wire or promise rejection and display via lightning-toast or custom UI.
-

55. How do you call a batch Apex job from LWC?

- Call an Apex @AuraEnabled method which enqueues the batch job.
-

56. How do you pass multiple parameters from LWC to Apex?

- Pass a **JavaScript object** and Apex accepts it as a Map or individual parameters.
-

57. How do you dynamically create multiple components in LWC?

- Use template `if:true` or `for:each` directives in template markup.
-

58. How do you handle CSS isolation in LWC?

- LWC uses **Shadow DOM**, so CSS in one component does not affect others.
-

59. How do you handle custom labels in LWC?

- Import labels using `@salesforce/label/c.LabelName` syntax.
-

60. How do you implement pagination in LWC?

- Use **current page index**, fetch limited records via Apex, and provide Next/Prev buttons.
-

61. How do you debug LWC?

- Use browser dev tools, console logs, Salesforce Lightning Inspector extension.
-

62. How do you handle picklist values in LWC?

- Fetch via **Apex** using `getPicklistValues` or `getPicklistValuesByRecordType` wire adapters.
-

63. How do you optimize LWC for large data sets?

- Lazy loading, pagination, cacheable=true in wire, avoid heavy DOM rendering.
-

64. How do you perform sorting in LWC datatable?

- Use onsort event to sort client-side or server-side.
-

65. How do you dynamically show/hide fields in LWC?

- Use conditional rendering with if:true or if:false.
-

66. How do you handle dependent picklists in LWC?

- Fetch controlling values via Apex and filter dependent values dynamically.
-

67. How do you call Flow from LWC?

- Use **Lightning Flow Screen Component** or lightning-flow LWC component.
-

68. How do you communicate between sibling LWCs?

- Use **custom events** or **Lightning Message Service**.
-

69. How do you handle modal dialogs in LWC?

- Use lightning-modal base component or custom modal with if:true conditional rendering.
-

70. How do you handle file upload in LWC?

- Use lightning-file-upload component and handle onuploadfinished event.
-

D. Visualforce (15 Q&A)

71. How do you call Apex from Visualforce page?

- Use `<apex:page controller="ControllerName">` and bind methods/variables using `{!methodName}`.
-

72. How do you pass parameters from Visualforce to Apex?

- Use **apex:param** with `<apex:commandButton>` or `<apex:actionFunction>`.
-

73. How do you rerender part of a VF page after action?

- Use rerender attribute on `<apex:commandButton>` or `<apex:actionFunction>`.
-

74. How do you call JavaScript from Apex in VF?

- Use `<apex:outputPanel>` with `oncomplete` or `actionFunction` callbacks.
-

75. How do you display a list of records in VF?

- Use `<apex:repeat>` or `<apex:dataTable>` bound to an Apex list variable.
-

76. How do you handle picklists in Visualforce?

- Use `<apex:selectList>` and populate options from Apex `List<SelectOption>`.
-

77. How do you implement pagination in VF?

- Maintain offset and limit variables in Apex controller and update on Next/Previous buttons.
-

78. How do you perform validation in VF page?

- Use Apex controller methods with `addError()` or `<apex:pageMessages>`.
-

79. How do you call Flow from Visualforce?

- Embed Flow using `<flow:interview name="FlowName"/>`.
-

80. How do you debug Visualforce pages?

- Use **debug logs**, inspect DOM, and add `System.debug` in Apex controller.
-

81. How do you use JavaScript remoting in VF?

- Annotate Apex methods with `@RemoteAction` and call via JavaScript using `Controller.methodName()`.
-

82. How do you implement dynamic styling in VF?

- Use `<apex:outputText styleClass="{!condition?'classA':'classB'}"/>`.
-

83. How do you handle large data sets in VF?

- Use **pagination** or **Apex StandardSetController**.
-

84. How do you secure VF pages?

- Set proper **profiles/permission sets** and enforce FLS/CRUD in Apex controller.
-

85. How do you handle file download in VF?

- Use `<apex:commandLink>` with `ContentType` and `getContent()` method in Apex.
-

86. How do you auto-assign Cases to agents?

- Use **Assignment Rules** or **Omni-Channel routing** based on criteria.

87. How do you prioritize Cases in Service Cloud?

- Use **Case Milestones**, **Entitlements**, or **Service Level Agreements (SLAs)**.
-

88. How do you send email notifications when Case status changes?

- Use **Workflow Rules**, **Process Builder**, or **Flow** to send emails.
-

89. How do you implement Omni-Channel in Salesforce?

- Configure **Presence Configurations**, **Routing Configurations**, and assign agents to **Queues**.
-

90. How do you prevent duplicate Cases?

- Enable **Duplicate Management rules** or use custom **Apex triggers**.
-

91. How do you implement Knowledge articles for Cases?

- Enable **Knowledge**, associate articles with Case record types, and use **Quick Actions** to link articles.
-

92. How do you escalate Cases automatically?

- Use **Escalation Rules** to reassign Cases or notify managers based on criteria.
-

93. How do you integrate chat with Service Cloud?

- Use **Live Agent / Omni-Channel Chat** with **LWC components** or embedded service SDK.
-

94. How do you handle case merging?

- Use **Merge Cases** standard functionality or custom Apex logic for specific criteria.
-

95. How do you automate Case closure based on resolution time?

- Use **Scheduled Flows** or **Time-Based Workflow** with criteria on Case Milestones.
-

 **F. Experience Cloud Scenarios (10 Q&A)**

96. How do you control portal user access?

- Use **Profiles, Permission Sets, and Sharing Rules** for Experience Cloud users.
-

97. How do you customize Experience Cloud pages?

- Use **Experience Builder, LWC components, and Themes.**
-

98. How do you expose Salesforce records to external users?

- Use **Sharing Sets, External Sharing, or Audience targeting** in Experience Cloud.
-

99. How do you implement single sign-on (SSO) for portal users?

- Use **SAML / OAuth SSO** with Identity Provider.
-

100. How do you implement self-service registration for external users?

- Use **Experience Cloud Registration Page** and assign default Profile/Permission Set.
-

101. How do you handle record visibility for external users?

- Use **Sharing Sets, Sharing Rules, and Manual Sharing** depending on the object.
-

102. How do you show dynamic content based on user profile?

- Use **Audience targeting** in Experience Builder.
-

103. How do you integrate Knowledge articles in Experience Cloud?

- Enable **Knowledge**, expose articles via **Components or LWC**, and control visibility.
-

104. How do you track user activity in portal?

- Enable **Event Monitoring, Reports, and Dashboards**.
-

105. How do you implement multi-language support in Experience Cloud?

- Enable **Translation Workbench** and provide translations for labels, articles, and components.
-

G. Security & Sharing Scenarios (5 Q&A)

106. How do you enforce field-level security in Apex?

- Use `Schema.sObjectField.isAccessible()` and `isUpdateable()` methods.
-

107. How do you enforce object-level security in Apex?

- Use `Schema.sObjectType.ObjectName.isAccessible()` before performing DML operations.
-

108. How do you programmatically share a record?

- Insert a record into the **Share** object (e.g., `AccountShare`, `CustomObject__Share`).

109. How do you prevent users from updating certain records?

- Use **Validation Rules**, **Trigger logic**, or **Apex Sharing** with readOnly access.

110. How do you implement row-level security for a large org?

- Use **Roles**, **OWD**, **Sharing Rules**, and **Criteria-Based Sharing** to control access.

 **H. Miscellaneous / Best Practices Scenarios (5 Q&A)**

111. How do you handle large data volumes in Salesforce?

- Use **Batch Apex**, **Async SOQL**, **Skinny Tables**, **selective queries**, and **Indexes**.

112. How do you optimize Apex code performance?

- Bulkify triggers, avoid unnecessary queries, cache frequently used data, and use collections efficiently.

113. How do you debug production issues without affecting users?

- Use **Debug Logs**, **Checkpoints**, **Developer Console**, and **Sandboxes** for testing.

114. How do you migrate metadata between orgs safely?

- Use **SFDX**, **Change Sets**, **Unlocked Packages**, **CI/CD pipelines** with validation deployments.

115. How do you ensure code is maintainable for future developers?

- Follow **Apex Enterprise Patterns**, use **Trigger Handlers**, comment code, write **unit tests**, and enforce **naming conventions**.

SALESFORCE INTERVIEW Q&A