Q-A
06

Apex Triggers Interview Q&A

4 of 28

1st) what is a Trigger?

→ This is a piece of particular event.

① Triggers are initiated when a record is inserted, updated, deleted and undeleted.

② We can perform custom operations before or after events to records.

③ Use triggers to do operations that cannot be done by point and clicks tools provided in Salesforce.

④ We can do things using triggers that we can do through Apex, including execution SOQL and DML or calling custom method.

⑤ Triggers can be created for both Standard & custom objects.

⑥ By default triggers are active as you create them.

1st) what is Trigger Syntax?

→

keyword — name of the trigger — keyword — sObject name

trigger triggerName on objectName (

event → triggerEvents) { // block of code

}

(65) Types of Triggers ?

→ (i) Before Trigger → it is used to update or validate record value before saved to database.

(ii) After Trigger → it is used to access field values that are set by the system such as Ids, and to make changes in the related / other records. The records that fire the after trigger are read only.

(66) what is this trigger events ?

→ (i) before insert
(ii) before update
(iii) before delete
(iv) after insert
(v) after update
(vi) after delete
(vii) after undelete

(66) what is Trigger Context variable ?

→ (i) isDelete → Returns true if trigger was fired due to a delete operation, from the Salesforce UI, Apex or API

(ii) isBefore → Returns true if the trigger was fired before any record was saved.

(iii) isAfter → Returns true if the trigger was fired after all record were saved.

(iv) isUndelete → Return true if the trigger was fired after a record is recovered from Recycle Bin.

(v) Size → The total number of records in a trigger invocation, both old & new.

(14) Trigger.new vs Trigger.newMap

→ new

(i) Returns a list of new version of sObject Records.

(ii) This sObject list is available in insert, update and undelete triggers, and the record can only be modified in before Triggers

newMap : →

(i) A Map of ids to the new versions of sObject records

(ii) Available in after insert, before update, after update, after undelete triggers.

(15) Trigger.old vs Trigger.oldMap ?

→ old

(i) Returns a list of old version of sObject records.

(ii) Available in before update, after update, before delete, after delete triggers.

oldMap :-

(i) A map of ids to the old versions of stored records.

(ii) Available in before update, after update before delete, after delete triggers

163) Can we call a apex class through trigger?

→ Yes, we can.

Example

trigger AccountTrigger on Account (before insert) {

AccountTriggerHandler.beforeInsert(Trigger.New);

}

public class AccountTriggerHandler {

public void beforeInsert (List<Account> newList) {

for (Account acc : Trigger.New) {

acc.Description = 'Test Description';

164) what is your Trigger Best practices ?

→ (i) One Trigger per object.

(ii) bulkify your code.

(iii) Logicless triggers.

(iv) Avoid using SOQL or DML inside for loop to avoid hitting governor limits.

(v) Avoid nested loops, try to use map instead

(vi) Use static boolean variable to avoid Recursive triggers.

(145) what is Recursive Trigger ?

→ ① In some scenarios it can happen that the result of the trigger can end up calling the same trigger again and again. This situation is known as recursive trigger.

② To avoid this scenario we should create a static variable and check the value of this variable before we execute anything in the trigger.

Example :

when you update a record from UI the trigger will be called. Now in trigger as well your apex update DML so it call same trigger again. only ends up as Recursion.

(146) Can we apply validation through trigger ?

→ yes, we can use addError() to apply validation through trigger.

(147) Can a trigger call a batch class ?

→ yes, we can call a batch class in the trigger as we do in the normal apex code.

(148) Can a trigger make a call Apex callout method ?

→ yes, we can call a callout method

in Apex Trigger but the only condition is that it has to be an asynchronous callout because the trigger flow cannot wait on the response received by the callout method.

(169) What is trigger Bulkification?

→ ① Trigger should be able to handle single records and bulk records.
② You should write triggers that operate on collections of sobjects.
③ Trigger should perform efficient SOQL and DML operations.

(170) Is there any limit on number of triggers defined on an object?

→ We can define as many triggers on an object but it is recommended to have one trigger per object as the order of execution of different trigger is not guaranteed and any trigger can start execution first.

(171) Order of execution in Triggers?

→ When you save a record with an insert, update or upsert statement. Salesforce performs the following events in order.

① Loads the original record from the database or initializes the record for an upsert statement.

(2) Loads the new record field values from the request and overwrites the old values.
(3) Executes record triggered flows that are configured to run before the record is saved.
(4) Executes all before triggers.
(5) Runs most system validation steps again and runs custom validation rules. The only system validation not redone doesn't run a second time (when the request comes from a standard UI edit page) is the enforcement of layout specific rules.
(6) Executes duplicate rules.
(7) Saves the record to the database but does not commit yet.
(8) Executes all after triggers.
(9) Executes assignment rules.
(10) Executes auto-response rules.
(11) Executes workflow rules.
(12) Executes escalation rules.
(13) Executes processes, flows launched by processes (order is not guaranteed)
(14) Executes entitlement rules.
(15) Executes record-triggered flows that are configured to run after the record is saved.

(16) If the record contains a roll-up summary field or is part of a cross-object workflow, performs calculations and updates the roll-up summary field in the parent record. Then parent record goes through save procedure.

(17) If the parent record is updated, and a grandparent record contains a roll-up summary field or is part of a cross-object workflows, performs calculations and updates the roll-up summary field in the grandparent record. Grandparent record goes through save procedure.

(18) Executes Criteria based sharing evaluation.

(19) Commits all DML operations to the database.

(20) After the changes are committed to the database, executes post-commit logic such as sending email and executing enqueued asynchronous Apex jobs, including queueable jobs and future methods.

Q-A
07

Apex Test class Q&A

- DML operations like Insert, update, Upsert, delete, undelete and Merge in Apex in Salesforce.

- Introduction :
  1. DML is used to insert, update, delete & undelete records.
  2. Use Upsert to either insert or update a record.
  3. Use merge when duplicate leads, Contacts and Accounts are there into one record. Others are deleted and related records are reparented.
  4. Always perform DML in Bulk
  5. Handle Exceptions

- Insert Records
  1. Insert Only one Records.
  2. Fetch it just after insertion of record.
  3. Insert records in bulk using List
  4. Governor limit check :
     - (i) Inserting two records Separately
     - (ii) Inserting two records through a List
  5. Insert related records. First insert Account then Contact.

Note: check newly created records in Org.

Update Records :

1. Query an existing record and print its values.
2. Then update the record and do update DML.
3. Again query the updated record.
4. Now apply System.assertEqual to validate update.
5. Update related records.

Note : check newly created / updated records in org

Upsert Records :

upsert means insert and update combining.

1. Create one record and update another one using Upsert DML.

Note → check newly created / updated record in Org.

Merge Records :

1. Only Lead, Contact and Account records can be merged.
2. You can merge upto three records of same sObject type

Note : → check newly created / updated records in Org.

Delete Records

① Apply SOQL
② Now delete queried records using Delete DML

Note → check deleted records in org.

undelete Records

① Insert a Record
② Delete that Record
③ Now query deleted records using ALL Rows in query.
④ Then apply undelete Statement.

Note → check deleted records in org.

DML Statement Exception

① Apply try & catch to handle exceptions raised when DML operation fails