APEX DEV

Created BY DHARMESH GUPTA

8/29/2022

Primitive Data Types

- Boolean
- String
- Integer
- Long
- Decimal
- Double
- Date
- Time
- Datetime
- Blob
- ID

Created BY DHARMESH GUPTA
8/29/2022

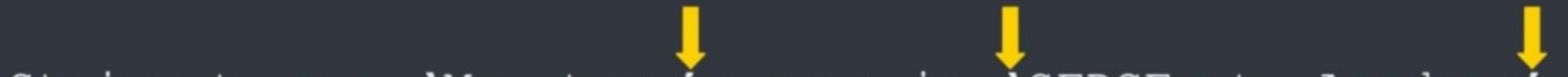
```
String greeting = 'Hello World';
System.debug(gree);

Boolean amIAwake = true;
System.debug(amIAwake);

Integer rollNumber = 11008890;
System.debug(rollNumber);

Long worldPopulation = 7000000000L;
System.debug(worldPopulation);

Decimal myCgpa = 8.5;
System.debug(myCgpa);

Double lightSpeed = 93000000/186000;
System.debug(lightSpeed);
```

Open Log | Execute | Execute Highlighted

```
String str = ' i am a string variable ';
System.debug('Actual String: '+str);

// capitalize string
System.debug('Capitalize String: '+str.capitalize());

// contains example
System.debug('Contains ring? :'+ str.contains('ring'));

// convert to upper case
System.debug('Upper case: '+str.toUpperCase());
// convert to lower case
System.debug('Lower case: '+str.toLowerCase());

// equals
System.debug('Is equal to ring?: '+str.equals('ring'));
String str1 = 'Manish';
String str2 = 'maNish';
System.debug('str1 equals str2: '+str1.equals(str2));
System.debug('str1 equals str2 ignore case: '+str1.toLowerCase().equals(str2.toLowe

// remove
System.debug('Remove ring: '+str.remove('ring'));

// replace
System.debug('Replace ring: '+str.replace('ring', 'rong'));

// split
```


Created BY DHARMESH GUPTA
8/29/2022

Escape characters in Strings



You can use the backslash character (\) to escape characters in a string.

Created BY DHARMESH GUPTA
8/29/2022

Escape Sequences

\b One backspace character

\n New line

\r Carriage return

\t Tab

\\ One backslash character

\o One ASCII null character

Created BY DHARMESH GUPTA
8/29/2022

Collections

Created BY DHARMESH GUPTA
8/29/2022

List

An ordered collection of elements of same datatype.

```
List<Integer> rollNumbers =
    new List<Integer>{11008890, 11008100,
11007231};
```

new Keyword

new keyword is used to create instance of a class and allocate memory.

List

Created BY DHARMESH GUPTA

8/29/2022

Index 0	Index 1	Index 2	....	Index n
11008890	11008100	11007231	....	89897767

Collections

Created BY DHARMESH GUPTA
8/29/2022

Set

An unordered collection of elements of same datatype.

```
Set<Integer> rollNumbers = new Set<Integer>{11008890, 11008100, 11007231};
```

OR

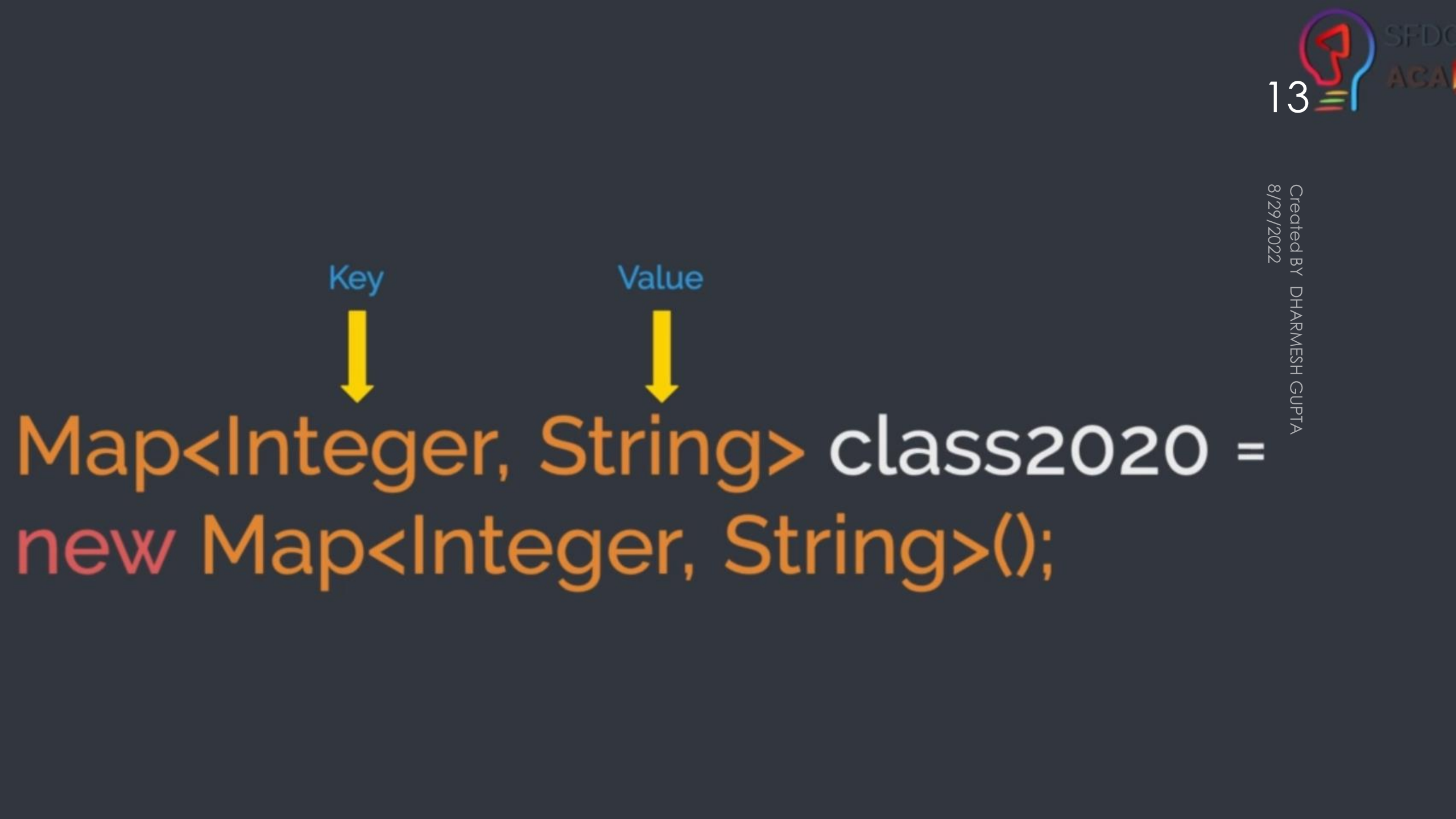
```
Set<Integer> rollNumbers = new Set<Integer>();
```

```apex
Set<Integer> rollNumbers = new Set<Integer>{11008890, 11008100, 11007231};
System.debug(rollNumbers);

rollNumbers.add(89897767);
rollNumbers.add(89897764);
rollNumbers.add(89897765);

System.debug(rollNumbers);

// adding duplicate values - NOT ALLOWED
rollNumbers.add(89897767);
System.debug(rollNumbers);

// check if set has an item
System.debug(rollNumbers.contains(89897764));
System.debug(rollNumbers.contains(345345));

// delete an item
rollNumbers.remove(89897765);
System.debug(rollNumbers);

// get set size
System.debug(rollNumbers.size());

// check if set is empty
System.debug(rollNumbers.isEmpty());
```


Created BY DHARMESH GUPTA
8/29/2022

Map

A key value pair based collection.



Created BY DHARMESH GUPTA
8/29/2022

SOQL Statement

Created BY DHARMESH GUPTA
8/29/2022

Multiple Conditions

Where clause can accept multiple conditions.

```
SELECT Name, Status, Company, Email FROM Lead

WHERE Status = 'Closed - Converted' OR Status = 'Closed - Not Converted'

WHERE Status = 'Closed - Converted' AND LeadSource = 'Web'
```

Created BY DHARMESH GUPTA
8/29/2022

IN Keyword

Add multiple possible values for a field

```
WHERE Status = 'Closed - Converted' OR Status = 'Closed - Not Converted'
```

```
WHERE Status IN ( 'Closed - Converted' , 'Closed - Not Converted' )
```

IN can give better performance in some circumstances but its not guaranteed and can only be confirmed by the query execution plan.

Created BY DHARMESH GUPTA
8/29/2022

LIKE Keyword

Match partial text string with support of wildcards

The % wildcard matches zero or more characters

LIKE 'fact%'

Matched Values
Fact
Facts
Factor

Not-matched
SFDCFacts
Facial

Created BY DHARMESH GUPTA
8/29/2022

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

Account | Lead@2:11 AM | Lead@2:13 AM | **Lead@2:15 AM**

```
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead ORDER BY CreatedDate DESC LIMIT 1
```

Query Results - Total Rows: 1

Name	Status	LeadSource	Company	Email	CreatedDate
Choudhary	Open - Not Contacted		SFDCFacts Academy		2020-06-10T20:39:21.000+0000

Query Grid: Save Rows | Insert Row | Delete Row | Refresh Grid

Access in Salesforce: Create New | Open Detail Page | Edit Page

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

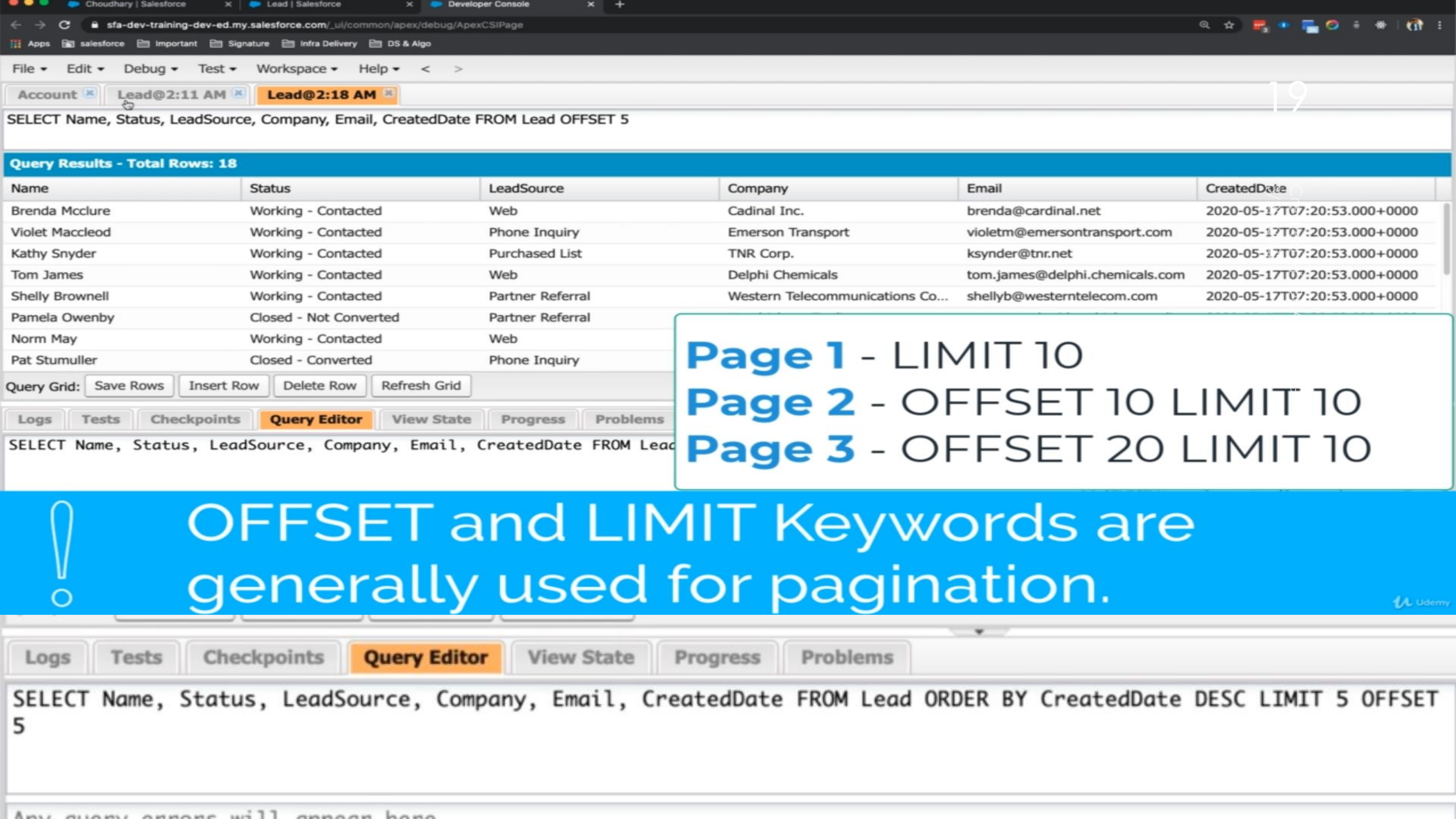
```
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead ORDER BY CreatedDate DESC LIMIT 1
```

Any query errors will appear here...

History

Executed

- SELECT Name, Status, LeadSource, Company, E...
- SELECT Name, Status, LeadSource, Company, E...
- SELECT Name, Status, LeadSource, Company, E...
- SELECT Name, Status, LeadSource, Company, E...


Choudhary | Salesforce
Lead | Salesforce
Developer Console
sfa-dev-training-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
Apps
salesforce
Important
Signature
Infra Delivery
DS & Algo
File
Edit
Debug
Test
Workspace
Help
Account
Lead@2:11 AM
Lead@2:18 AM
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead OFFSET 5
Query Results - Total Rows: 18
Name
Status
LeadSource
Company
Email
CreatedDate
Brenda Mcclure
Working - Contacted
Web
Cadinal Inc.
brenda@cardinal.net
2020-05-17T07:20:53.000+0000
Violet Maccleod
Working - Contacted
Phone Inquiry
Emerson Transport
violetm@emersontransport.com
2020-05-17T07:20:53.000+0000
Kathy Snyder
Working - Contacted
Purchased List
TNR Corp.
ksynder@tnr.net
2020-05-17T07:20:53.000+0000
Tom James
Working - Contacted
Web
Delphi Chemicals
tom.james@delphi.chemicals.com
2020-05-17T07:20:53.000+0000
Shelly Brownell
Working - Contacted
Partner Referral
Western Telecommunications Co...
shellyb@westerntelecom.com
2020-05-17T07:20:53.000+0000
Pamela Owenby
Closed - Not Converted
Partner Referral
Norm May
Working - Contacted
Web
Pat Stumuller
Closed - Converted
Phone Inquiry
Query Grid:
Save Rows
Insert Row
Delete Row
Refresh Grid
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead
19
Page 1 - LIMIT 10
Page 2 - OFFSET 10 LIMIT 10
Page 3 - OFFSET 20 LIMIT 10
OFFSET and LIMIT Keywords are generally used for pagination.
Udemy
Logs
Tests
Checkpoints
Query Editor
View State
Progress
Problems
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead ORDER BY CreatedDate DESC LIMIT 5 OFFSET 5

SOQL Operators

Equals (=)	`Name = 'SFDCFacts Academy'`
Not Equals (!=)	`Name != 'Peter'`
Less than (<)	`Amount < 1000`
Less than or equal (<=)	`Amount <= 1000`
Greater than (>)	`Date > '2020-06-11'`
Greater than or equal (>=)	`Date >= '2020-06-11'`
LIKE	`Name LIKE 'SFDCFacts%'`
IN	`Name IN ('SFDCFacts', 'SFDCFacts Academy')`
NOT IN	`Name NOT IN ('Harry', 'Ron')`

Created BY DHARMESH GUPTA
8/29/2022

Date Literals

You can use date literals to compare the range of values.

TODAY - Starts at 00:00:00 of current day

YESTERDAY - Starts at 00:00:00 the day before

TOMORROW - Starts at 00:00:00 after the current day

Created BY DHARMESH GUPTA
8/29/2022

Date Literals

LAST_WEEK	LAST_MONTH	LAST_YEAR	LAST_QUARTER
THIS_WEEK	THIS_MONTH	THIS_YEAR	THIS_QUARTER
NEXT_WEEK	NEXT_MONTH	NEXT_YEAR	NEXT_QUARTER

LAST_N_DAYS:n	LAST_N_MONTHS:n
NEXT_N_DAYS:n	NEXT_N_MONTHS:n
LAST_N_WEEKS:n	LAST_N_YEAR:n
NEXT_N_WEEKS:n	NEXT_N_YEAR:n

LAST_90_DAYS

NEXT_90_DAYS

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

```
SELECT Name, Status, LeadSource, Company, Email, CreatedDate FROM Lead WHERE CreatedDate=LAST_N_DAYS:10
```

Any query errors will appear here...

Created BY DHARMESH GUPTA
8/29/2022

```
// Step 1: Write a SOQL query to retrieve all Contacts from a Salesforce Org. Retrieved results should show Contact Name, Title, Phone
SELECT Name, Title, Phone, Email FROM Contact

// Step 2: Modify SOQL query to retrieve only those contacts with title 'VP, Technology'.
SELECT Name, Title, Phone, Email FROM Contact WHERE Title='VP, Technology'

// Step 3: Modify SOQL, and add another field called 'Department' in the results. This field should be the 2nd field in the result.
SELECT Name, Department, Title, Phone, Email FROM Contact WHERE Title='VP, Technology'

// Step 4: Modify SOQL, retrieve all results satisfying step 2 condition and has department value as 'Finance'
SELECT Name, Department, Title, Phone, Email FROM Contact WHERE Title='VP, Technology' AND Department='Finance'

// Step 5: Modify SOQL, include all SVP and VP in your search results. Make sure your result still meets step 4 condition.
SELECT Name, Department, Title, Phone, Email FROM Contact WHERE Title LIKE '%VP%' AND Department='Finance'

// Step 6: Sort results by name in descending order.
SELECT Name, Department, Title, Phone, Email FROM Contact WHERE Title LIKE '%VP%' AND Department='Finance' ORDER BY Name DESC

// Step 7: Limit search result to only 3.
SELECT Name, Department, Title, Phone, Email FROM Contact WHERE Title LIKE '%VP%' AND Department='Finance' ORDER BY Name DESC LIMIT 3
```

```
SELECT Name,
    RegistrationNumber,
    (SELECT Name,
        RollNumber
    FROM Students)
FROM School
```

SELECT Name, RegistrationNumber, (SELECT Name, RollNumber FROM Students) FROM School

File Edit Debug Test Workspace Help < >

Account@10:24 PM | Account@10:25 PM | Account@10:26 PM | Account@10:27 PM | Account@10:28 PM | Account@10:29 PM | **Account@10:31 PM**

```
SELECT Name, Phone, Website, (SELECT Name, Department, Email FROM Contacts WHERE Department='Finance'), (SELECT Name, Amount FROM Opportunities) FROM Account WHERE Name='United Oil & Gas, UK'
```

Query Results - Total Rows: 1

Name	Phone	Website	Contacts	Opportunities
United Oil & Gas, UK	+44 191 4956203	http://www.uos.com	[{"Name":"Ashley James","Department"...	

Query Grid: Save Rows | Insert Row | Delete Row | Refresh Grid

Access in Salesforce: Create New | Open Detail Page | Edit Page

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

```
SELECT Name, Phone, Website, (SELECT Name, Department, Email FROM Contacts WHERE Department='Finance'), (SELECT Name, Amount FROM Opportunities) FROM Account
```

History

Executed

SELECT Name, Department, Title, Phone, Email ...

Query Grid: Save Rows | Insert Row | Delete Row | Refresh Grid

Access in Salesforce: Create New | Open Detail Page | Edit Pag

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

```
SELECT Name, (SELECT Name FROM Books__r) FROM Author__c
```

History

Executed

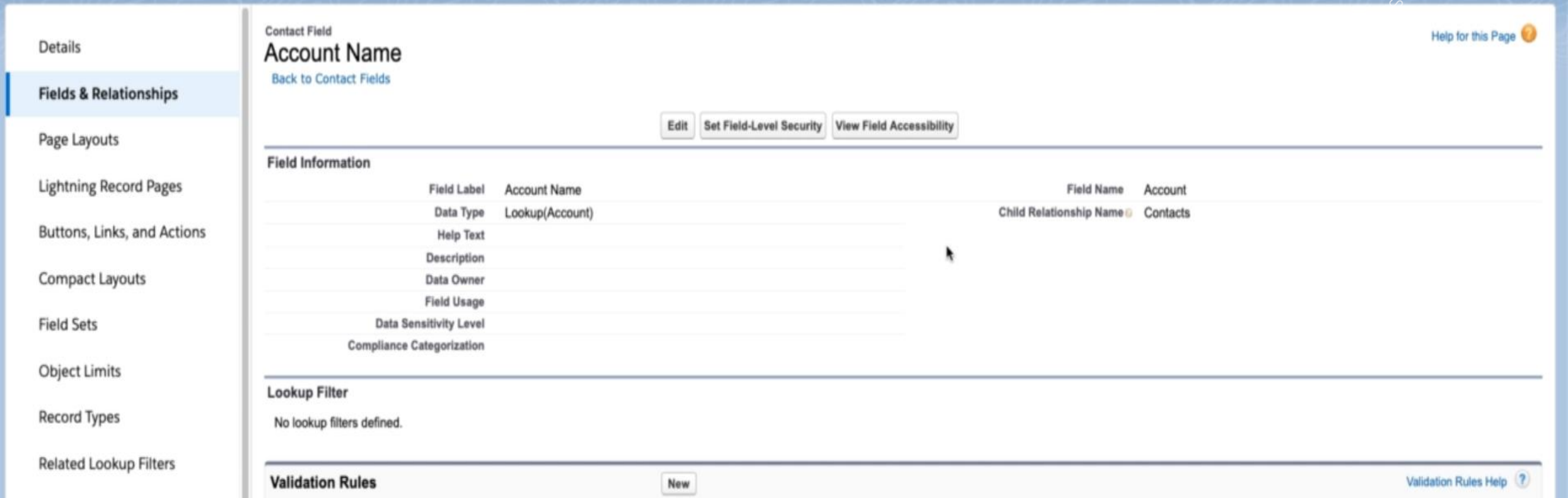
SELECT Name FROM Account

! Custom object relationship name must have a suffix __r.

Parent to Child Query - Limitations

Only one level of parent-to-child is supported.

Upto 20 related objects are supported.



Created BY DHARMESH GUPTA
8/29/2022

```
SELECT Name,
       RollNumber,
       School.Name,
       School.RegistrationNumber
FROM Student
```

SELECT Name, RollNumber, School.Name, School.RegistrationNumber FROM Student

Child to Parent Query - Limitations

Only five level of child-to-parent is supported.

Upto 55 related objects are supported.

Created BY DHARMESH GUPTA
8/29/2022

```
// Contact is a parent object of case
// Account is a parent object of Contact

// Retrieve Name, Department and Title of all contacts
SELECT Name, Department, Title FROM Contact

// Retrieve all Cases (CaseNumber, Subject) raised by the contact
SELECT Name, Department, Title, (Select CaseNumber, Subject FROM Cases) FROM Contact

// Get parent Account's Name, Rating for each Contact
SELECT Name, Department, Title, (Select CaseNumber, Subject FROM Cases), Account.Name, Account.Rating FROM Contact

// Make sure Account fields are the initial columns in results
SELECT Account.Name, Account.Rating, Name, Department, Title, (Select CaseNumber, Subject FROM Cases) FROM Contact

// Retrieve only those records where Account Rating is Hot
SELECT Account.Name, Account.Rating, Name, Department, Title, (Select CaseNumber, Subject FROM Cases) FROM Contact
WHERE Account.Rating='Hot'

// Sort results by Contact Name
SELECT Account.Name, Account.Rating, Name, Department, Title, (Select CaseNumber, Subject FROM Cases) FROM Contact
WHERE Account.Rating='Hot' ORDER BY Name

// Only retrieve open cases (use IsClosed Checkbox field value)
SELECT Account.Name, Account.Rating, Name, Department, Title, (Select CaseNumber, Subject FROM Cases
 WHERE IsClosed=false) FROM Contact WHERE Account.Rating='Hot' ORDER BY Name

// Add one more filter condition, Contact Department must be equals to 'Technology'
SELECT Account.Name, Account.Rating, Name, Department, Title, (Select CaseNumber, Subject FROM Cases
    WHERE IsClosed=false) FROM Contact WHERE Account.Rating='Hot' AND Department='Technology' ORDER BY Name
```

```apex
// Retrieve all accounts and assign to a List collection
List<Account> accounts = [SELECT Name, Phone FROM Account];

// iterate over all accounts and print account information
for(Account account : accounts){
    System.debug('Account Name: '+account.Name+' Account Phone: '+account.Phone);
}

// Get SOQL results in a Map collection
Map<Id, Account> accountsMap = new Map<Id, Account>([SELECT Name, Phone FROM Account]);
for(Account account : accountsMap.values()){
    System.debug('Account Name: '+account.Name+' Account Phone: '+account.Phone);
}

// Retrieve all books and assign to a List collection
List<Book__c> books = [SELECT Name, Price__c FROM Book__c];
for(Book__c book : books){
    System.debug('Book Name: '+book.Name+' Book Price: '+book.Price__c);
}

```

Created BY DHARMESH GUPTA
8/29/2022

Binding Variables in SOQL

You can use Apex variables in SOQL using a colon (:). These variables are called bind variables.

```
Set<Id> accountIds = new Set<Id>();
[SELECT Id, Name, Rating FROM Account WHERE Id IN :accountIds];

String leadName = 'XYZ Corp';
[SELECT Id, Name FROM Lead WHERE Name =:leadName];
```

Binding Variables in SOQL

Bind expressions can be used as:

- The filter literals in WHERE clauses.
- The value of the IN or NOT IN operator in WHERE clauses, allowing filtering on a dynamic set of values. Note that this is of particular use with a list of IDs or Strings, though it works with lists of any type.
- The numeric value in LIMIT clauses.
- The numeric value in OFFSET clauses.

Dynamic SOQL Queries

You can build your SOQL queries at runtime in apex code dynamically. Dynamic SOQL enables you to create more flexible applications.

```
String dynQuery = 'SELECT Id, Name FROM Account';
if(accountType == 'Red'){
    dynQuery += 'WHERE Rating=\'Hot\' AND Amount > 100,000';
}

List<Account> accounts = Database.query(dynQuery);
```

Database Methods

```
Database.SaveResult[] srList =
Database.insert(accList, false);
```

You can control partial record processing

NO exception is thrown, instead a result array is returned

DML Statements


```
insert accList;
```

NO partial record processing

Results in exception

Created BY DHARMESH GUPTA
8/29/2022

Primitive Data Types

- Boolean
- String
- Integer
- Long
- Decimal
- Double
- Date
- Time
- Datetime
- Blob
- ID

Created BY DHARMESH GUPTA
8/29/2022

Collections

List

Set

Map

Class as Datatype

```
Covid19 jaipur = new Covid19();
```

Datatype — Variable

```
List<Covid19> hpCities = new List<Covid19>();

Set<Covid19> hpCities = new Set<Covid19>();

Map<String, List<Covid19>> stateToCityMap  = new Map<String,
List<Covid19>>();
```

Standard/Custom Object as Datatype


```
List<Account> accounts = [SELECT Name, Phone FROM Account];
```

```
Map<Id, Account> accountsMap = new Map<Id, Account>([SELECT Name,
Phone FROM Account]);
```

Created BY DHARMESH GUPTA
8/29/2022

SObject

Any standard or custom object is an SObject.

- Account
- Contact
- Lead
- Book__c
- Author__c

Created BY DHARMESH GUPTA
8/29/2022

SObject


```
List<SObject> accounts = [SELECT Id, Name FROM Account];

List<SObject> contacts = [SELECT Id, Name FROM Contact];

List<SObject> leads = [SELECT Id, Name FROM Lead];

List<SObject> books = [SELECT Id, Name FROM Book__c];

List<SObject> authors = [SELECT Id, Name FROM Author__c];
```

Created BY DHARMESH GUPTA
8/29/2022

Created BY DHARMESH GUPTA
8/29/2022

```
/**
 * DML CHALLENGE
 * Create a new Account with the name “WingNut Films” and Rating “Warm”
 * Create a new Opportunity “Lord Of The Rings”  for this Account with below field values
 *                Stage > Qualification
 *                Closed Date > Today
 * Update Account Name to “New Line Cinema”
 * Update Opportunity Stage to “Closed-Won”
 * Delete the opportunity
 * Undelete the opportunity
 **/

// create new instance of account object
Account accRec = new Account(Name='WingNut Films', Rating='Warm');
// insert the account
insert accRec;


// create opportunity record
Account account = [SELECT Id FROM Account WHERE Name='WingNut Films' LIMIT 1];
Opportunity opp = new Opportunity(Name='Lord Of The Rings', StageName='Qualification ', CloseDate=Date.today());
opp.AccountId = account.Id;
insert opp;

// retrieve wingnut films account
Account account = [SELECT Id, Name FROM Account WHERE Name='WingNut Films' LIMIT 1];
// update the name
account.Name = 'New Line Cinema';
// update the account
update account;
```

```
Opportunity opp = [SELECT Id, Name, StageName FROM Opportunity WHERE Name='Lord Of The Rings' LIMIT 1];
opp.StageName = 'Closed Won';
update opp;

// get opportunity record to delete
Opportunity opp = [SELECT Id, Name, StageName FROM Opportunity WHERE Name='Lord Of The Rings' LIMIT 1];
// delete opportunity
delete opp;

// get opportunity record to undelete
List<Opportunity> opps = [SELECT Id, Name, StageName FROM Opportunity WHERE isDeleted=true ALL ROWS];
// undelete opportunity
undelete opps;
```

Exception

Apex uses exceptions to note errors and other events that disrupt the normal flow of code execution.

Use **throw** keyword to generate an exception programmatically

Handling an Exception

The try, catch, and finally statements can be used to gracefully recover from a thrown exception.

```
System.debug('Before Exception');
try{
    Integer i = 10/0; // throws an
exception
} catch(MathException e){
    // catch exception here
} finally {
    // optional finally block
}
System.debug('After Exception');
```

Catch vs Finally

Catch block only executes if the exception was thrown in try block and matches it's exception type.

Finally block always executes regardless of whether an exception was thrown.

Apps | salesforce | Signature | Important | DS & Algo | Resources | Infra Delivery | VAC | DCEng-GLOBAL -...

Edit | Debug | Test | Workspace | Help | < >

Log executeAnonymous @17/07/2020

Enter Apex Code

```
1  System.debug('Before Exception');
2  try{
3      // Math Exception
4      Double result = 10/0;
5      System.debug('Result: '+result);
6
7      // DML Exception
8      // Account accRec = new Account();
9      // insert accRec;
10     System.debug('Empty try block ');
11 } catch(Exception e){
12     System.debug('Cought generic exception ');
13 } finally {
14     System.debug('finally called ');
15 }
16 System.debug('After Exception');
```

Open Log | Execute | Execute Highlighted

Execution Log

Timestamp	Event	Details
0:35:17:003	USER_DEBUG	[9]|DE
0:35:17:003	USER_DEBUG	[13]|D

This Frame | Executable | Debug C

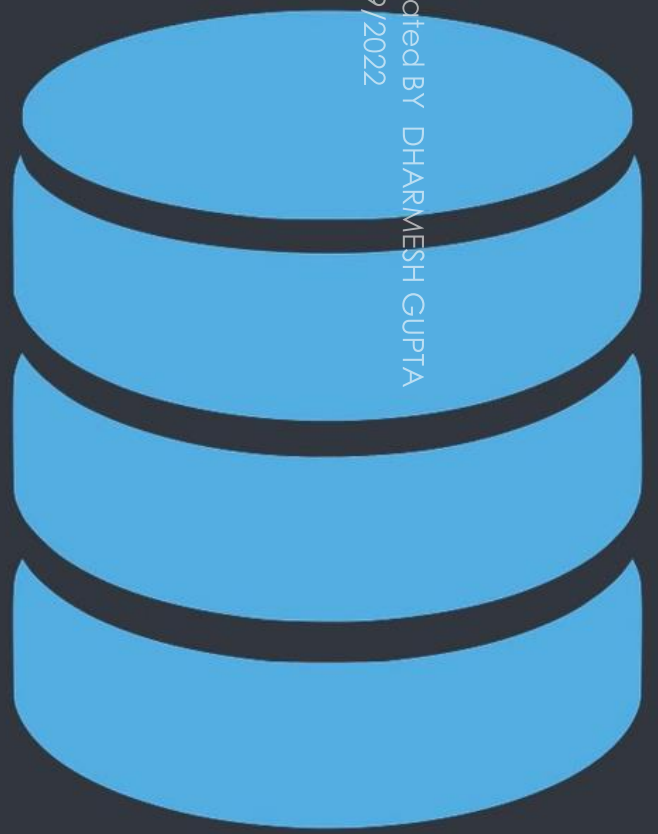
Logs | Tests | Checkpoints | Query

User	Application				Size
anish Choudhari	Unknown	/services/data/v48.0/tooling/ex...	17/07/2020, 00:35:17	Success	2.67 KB
anish Choudhari	Unknown	/services/data/v48.0/tooling/ex...	17/07/2020, 00:34:15	Success	4.52 KB
anish Choudhari	Unknown	/services/data/v48.0/tooling/ex...	17/07/2020, 00:32:09	Success	4.63 KB
anish Choudhari	Unknown	/services/data/v48.0/tooling/ex...	17/07/2020, 00:31:53	Success	4.64 KB

What is database trigger?

A database trigger is special set on operations that is run when specific actions occur within a database.

Triggers are defined to run when changes are made to a table's data.



What is Apex trigger?

An apex trigger is apex code that is run when specific actions occur within a Salesforce object.

Apex Triggers are defined to run when changes are made to an object's records.

Created BY DHARMESH GUPTA
8/29/2022

Apex Trigger

Apex supports triggers before and after following events:

- Insert
- Update
- Delete
- Merge
- Upsert
- Undelete

Created BY DHARMESH GUPTA
8/29/2022

Trigger Syntax

Trigger Name

SObject Name

```
trigger myTestTrigger on Lead (before insert, after update) {

        // Trigger code here
        // you can write as many lines
        // of codes as you want here
}
```

Trigger Context Variables

isInsert - Return true if this trigger was fired due to an insert operation.

isUpdate - Return true if this trigger was fired due to an update operation.

isDelete - Return true if this trigger was fired due to a delete operation.

isUndelete - Return true if this trigger was fired due to an undelete operation.

Trigger Context Variables

new - List of new versions of records. Available in [before insert, after insert, before update, after update, after undelete]

Trigger.new is same as List<SObject>

newMap - Map of Id and new versions of records. Available in [after insert, before update, after update, after undelete]

Trigger.newMap is same as Map<Id, SObject>

Trigger Context Variables

isExecuting - Return true if current context for the Apex code is a trigger.

size - The total number of records in a trigger invocation, both old and new.

operationType - Return an enum corresponding to the current operation. Possible values are: [BEFORE_INSERT, AFTER_INSERT, BEFORE_UPDATE, AFTER_UPDATE, BEFORE_DELETE, AFTER_DELETE, AFTER_UNDELETE]

Trigger Context Variables

isExecuting - Return true if current context for the Apex code is a trigger

size - The total number of records in a trigger invocation, both old and new.

operationType - Return an enum corresponding to the current operation.
Possible values are: [BEFORE_INSERT, AFTER_INSERT, BEFORE_UPDATE, AFTER_UPDATE, BEFORE_DELETE, AFTER_DELETE, AFTER_UNDELETE]

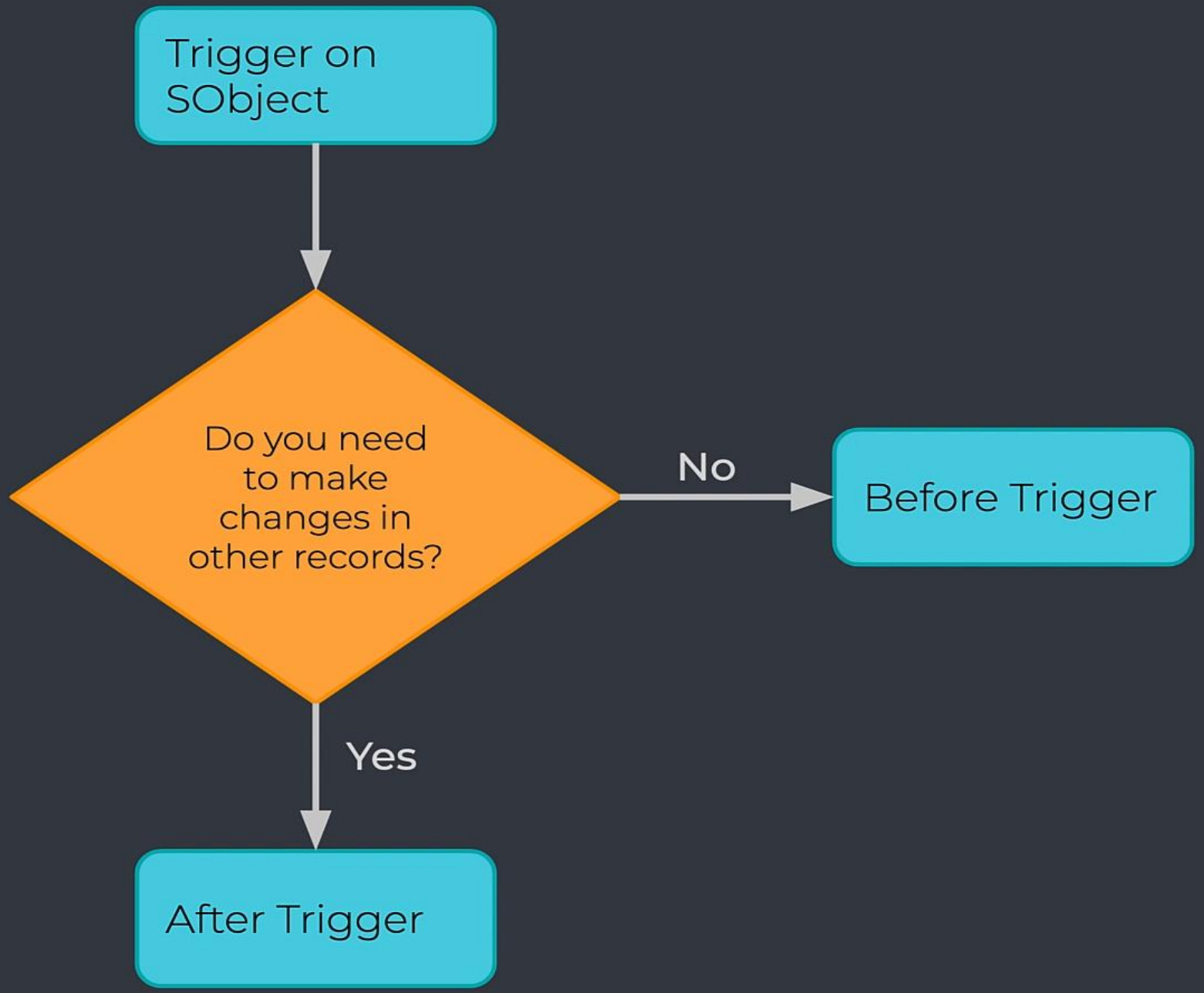
Before vs After

Before triggers are used to update or validate record values before they're saved to the database.

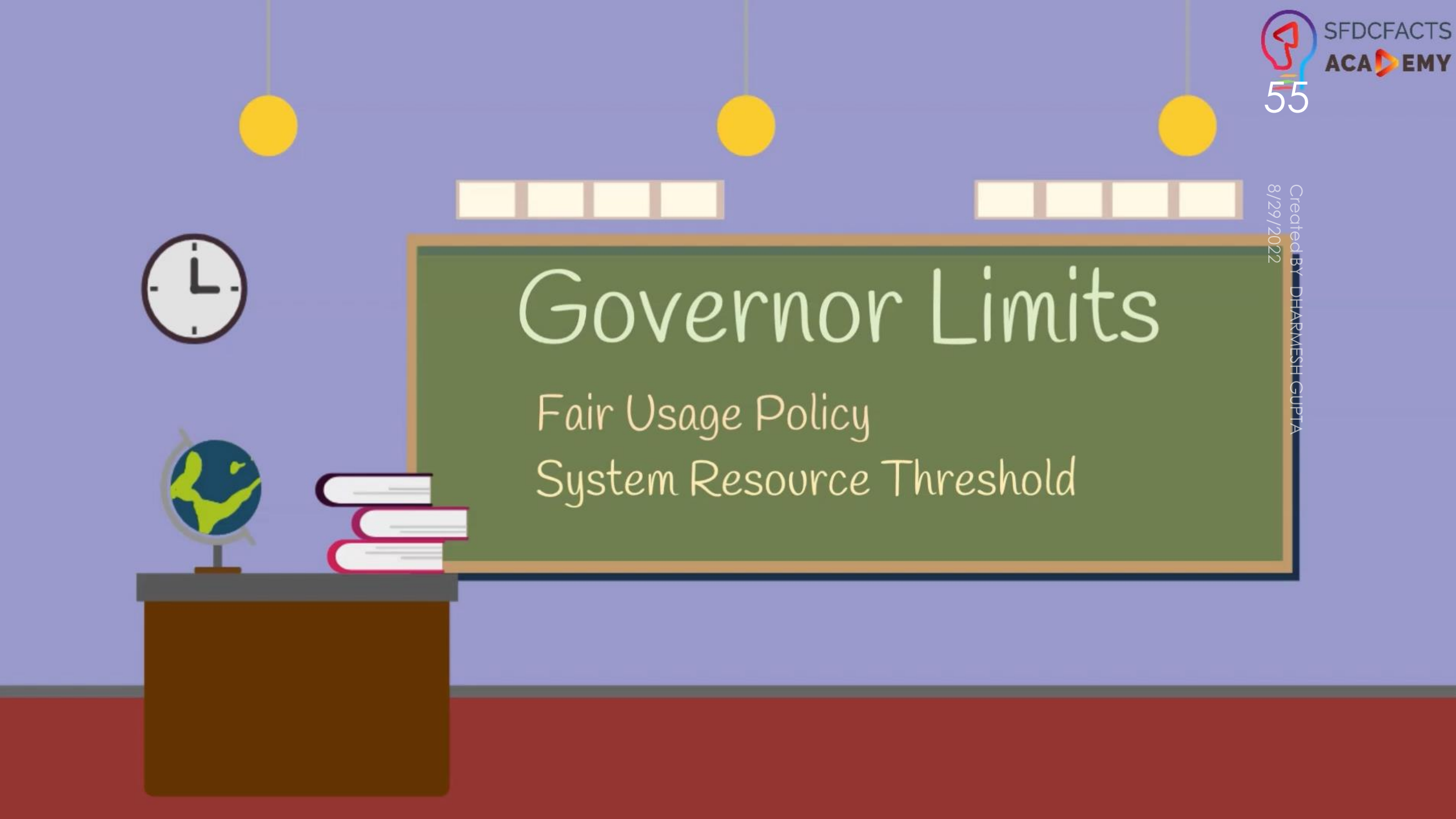
Avoid making changes to other records in before triggers.

Records that fire the after trigger are read only.

Make changes (create/update/delete) to other records in after triggers.



Created BY DHARMESH GUPTA
8/29/2022



Created BY- DHARMESH GUPTA
8/29/2022

d Deploying

nce

Description	Synchronous Limit	Asynchronous Limit
Total number of SOQL queries issued[1]	100	200
Total number of records retrieved by SOQL queries	50,000	
Total number of records retrieved by `Database.getQueryLocator`	10,000	
Total number of SOSL queries issued	20	
Total number of records retrieved by a single SOSL query	2,000	
Total number of DML statements issued[2]	150	
Total number of records processed as a result of DML statements, `Approval.process`, or `database.emptyRecycleBin`	10,000	
Total stack depth for any Apex invocation that recursively fires triggers due to `insert`, `update`, or `delete` statements[3]	16	
Total number of callouts (HTTP requests or web services calls) in a transaction	100	
Maximum cumulative timeout for all callouts (HTTP requests or Web services calls) in a transaction	120 seconds	
Maximum number of methods with the `future` annotation allowed per Apex invocation	50	0 in batch and future contexts; 1 in queueable context

Bulkification

Consume least amount of resources

Less chances of hitting governor limits

Reduces transaction execution time

Created BY DHARMESH GUPTA
8/29/2022

NEVER WRITE SOQL QUERIES AND DML STATEMENTS IN LOOPS

```
Opportunity opp = [SELECT Id, Amount, Profile__c FROM Opportunity WHERE Id =: oppId];
Opportunity opp = [SELECT Id, Amount, Profile__c FROM Opportunity WHERE Id IN: oppId];
for(Id oppId : oppIds){
```

Limits Class

You can get the current limit usage in apex using methods in Limits class.

getDMLStatements() - Returns the number of DML statements (such as insert, update or the database.EmptyRecycleBin method) that have been called.

getHeapSize() - Returns the approximate amount of memory (in bytes) that has been used for the heap.

Limits Class

You can get the allowed limits in apex using methods in Limits class.

Created BY DHARMESH GUPTA
8/29/2022

getLimitDMLStatements() - Returns the total number of DML statements or the database.EmptyRecycleBin methods that can be called.

getLimitHeapSize() - Returns the total amount of memory (in bytes) that can be used for the heap.

See Execution Governors and Limits.

Limits Methods

The following are methods for `Limits`. All methods are static.

- getAggregateQueries()
  Returns the number of aggregate queries that have been processed with any SOQL query statement.
- getLimitAggregateQueries()
  Returns the total number of aggregate queries that can be processed with SOQL query statements.
- getAsyncCalls()
  Reserved for future use.
- getLimitAsyncCalls()
  Reserved for future use.
- getCallouts()
  Returns the number of Web service statements that have been processed.
- getChildRelationshipsDescribes()
  Deprecated. Returns the number of child relationship objects that have been returned.
- getLimitCallouts()
  Returns the total number of Web service statements that can be processed.
- getCpuTime()
  Returns the CPU time (in milliseconds) that has been used in the current transaction.
- getLimitCpuTime()
  Returns the maximum CPU time (in milliseconds) that can be used in a transaction.
- getDMLRows()
  Returns the number of records that have been processed with any statement that counts against DML limits, such as DML statements, the `Database.emptyRecycleBin` method, and other methods.
- getLimitDMLRows()
  Returns the total number of records that can be processed with any statement that counts against DML limits, such as DML statements, the `database.EmptyRecycleBin` method, and other methods.
- getDMLStatements()
  Returns the number of DML statements (such as `insert`, `update` or the `database.EmptyRecycleBin` method) that have been called.
- getLimitDMLStatements()

Aggregate Functions

AVG() - Returns the average value of a numeric field

COUNT() and **COUNT(fieldName)** - Return number of rows

COUNT_DISTINCT() - Returns the number of distinct non-null field values

MIN() - Returns the minimum value of a field.

MAX() - Returns the maximum value of a field

SUM() - Returns the total sum of a numeric field

GROUP BY CLAUSE

You can use the GROUP BY option in a SOQL query to specify a group of records instead of processing many individual records.

Number of Leads associated with each Lead Source
SELECT LeadSource, COUNT(Id) FROM Lead GROUP BY LeadSource

Max opportunity amount in each stage
SELECT StageName, MAX(Amount) FROM Opportunity GROUP BY StageName

HAVING CLAUSE

HAVING is an optional clause that can be used in a SOQL query to filter results that aggregate functions return

Group by StageName where Sum of amount is greater than 500k

```
SELECT StageName, SUM(Amount) FROM Opportunity GROUP BY StageName HAVING SUM(AMOUNT)>500000
```

Opportunity@9:39 PM | Opportunity@9:39 PM | Opportunity@9:40 PM

```
SELECT StageName, Sum(Amount) FROM Opportunity WHERE Amount>300000 GROUP BY StageName
```

Query Results - Total Rows: 2

StageName	sum(Amount)
Needs Analysis	675000
Closed Won	1705000

Query Grid: Save Rows | Insert Row | Delete Row | Refresh Grid

Access in Sa

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

```
SELECT StageName, Sum(Amount) FROM Opportunity WHERE Amount>300000 GROUP BY StageName
```

File Edit Debug Test Workspace Help < >

Opportunity@9:39 PM | Opportunity@9:39 PM

```
SELECT StageName, Sum(Amount) FROM Opportunity GROUP BY StageName HAVING Sum(Amount) > 300000
```

Query Results - Total Rows: 5

StageName	sum(Amount)
Needs Analysis	675000
Value Proposition	330000
Proposal/Price Quote	370000
Negotiation/Review	395000
Closed Won	3645000

Query Grid: Save Rows | Insert Row | Delete Row | Refresh Grid

Access in Salesforce: Create New | Open Detail Page | Edit Page

Logs | Tests | Checkpoints | **Query Editor** | View State | Progress | Problems

```
SELECT StageName, Sum(Amount) FROM Opportunity GROUP BY StageName HAVING Sum(Amount) > 300000
```

Any query errors will appear here...

History

Executed
SELECT StageName, SUM(Amount) FROM Oppor...
SELECT StageName, SUM(Amount) FROM Oppor...
SELECT StageName, SUM(Amount) FROM Oppor...
SELECT StageName, SUM(Amount) FROM Oppor...

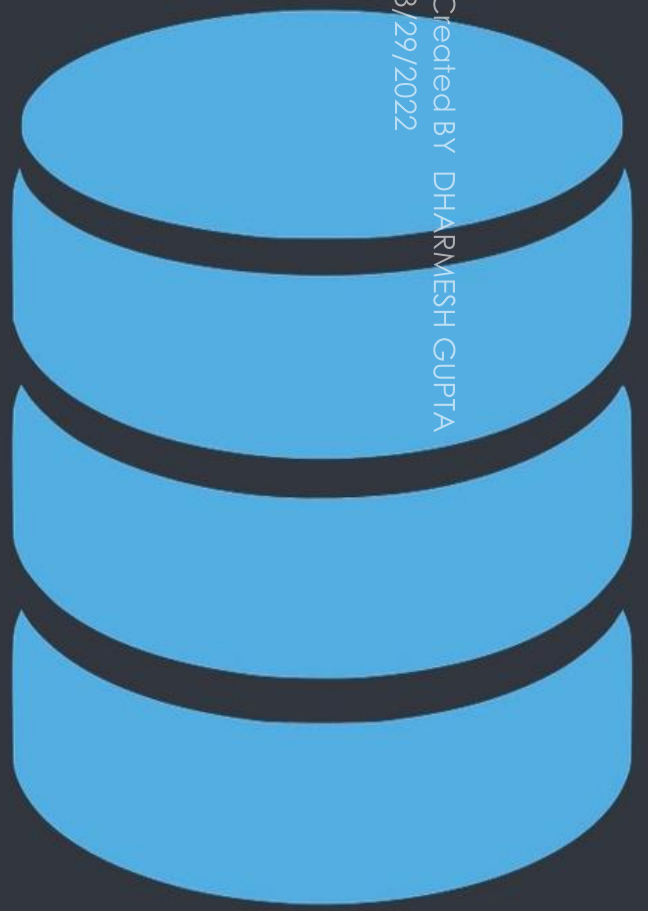
Aggregate Functions


```
AggregateResult[] groupedResults

  = [SELECT AVG(Amount), MAX(Amount) FROM Opportunity];



Double avgAmount = Double.valueOf(groupedResults[0].get('expr0'));

Double maxAmount = Double.valueOf(groupedResults[0].get('expr1'));
```

What is database trigger?

A database trigger is special set on operations that is run when specific actions occur within a database.

Triggers are defined to run when changes are made to a table's data.



Trigger Syntax

Trigger Name

SObject Name

```
trigger myTestTrigger on Lead (before insert, after update) {

        // Trigger code here
        // you can write as many lines
        // of codes as you want here
}
```


Why we need Apex Trigger?

Triggers are extremely helpful in automating business processes.

Why we need Apex Trigger?

Triggers give you endless possibilities in automating your tasks.

Triggers can be used to implement complex validation.

Triggers can be used to communicate with third-party systems when data changes in Salesforce.

Triggers are extremely fast.

Created BY DHARMESH GUPTA
8/29/2022

Trigger Events

You can define trigger for following events

before insert
after insert

before update
after update

before delete
after delete

after undelete

Validations and throwing errors

Triggers can be used to prevent DML operations from occurring by calling the **addError()** method on a record or field

Trigger Context Variables

isInsert – Return true if this trigger was fired due to an insert operation.

isUpdate – Return true if this trigger was fired due to an update operation.

isDelete – Return true if this trigger was fired due to a delete operation.

isUndelete – Return true if this trigger was fired due to an undelete operation.

isBefore – Return true if this trigger was fired before any record was saved.

isAfter – Return true if this trigger was fired after all record were saved.

Trigger Context Variables

new - List of new versions of records. Available in [before insert, after insert, before update, after update, after undelete]

Trigger.new is same as List<SObject>

newMap - Map of Id and new versions of records. Available in [after insert, before update, after update, after undelete]

Trigger.newMap is same as Map<Id, SObject>

Trigger Context Variables

isExecuting - Return true if current context for the Apex code is a trigger.

size - The total number of records in a trigger invocation, both old and new.

operationType - Return an enum corresponding to the current operation.

Possible values are: [BEFORE_INSERT, AFTER_INSERT, BEFORE_UPDATE, AFTER_UPDATE, BEFORE_DELETE, AFTER_DELETE, AFTER_UNDELETE]

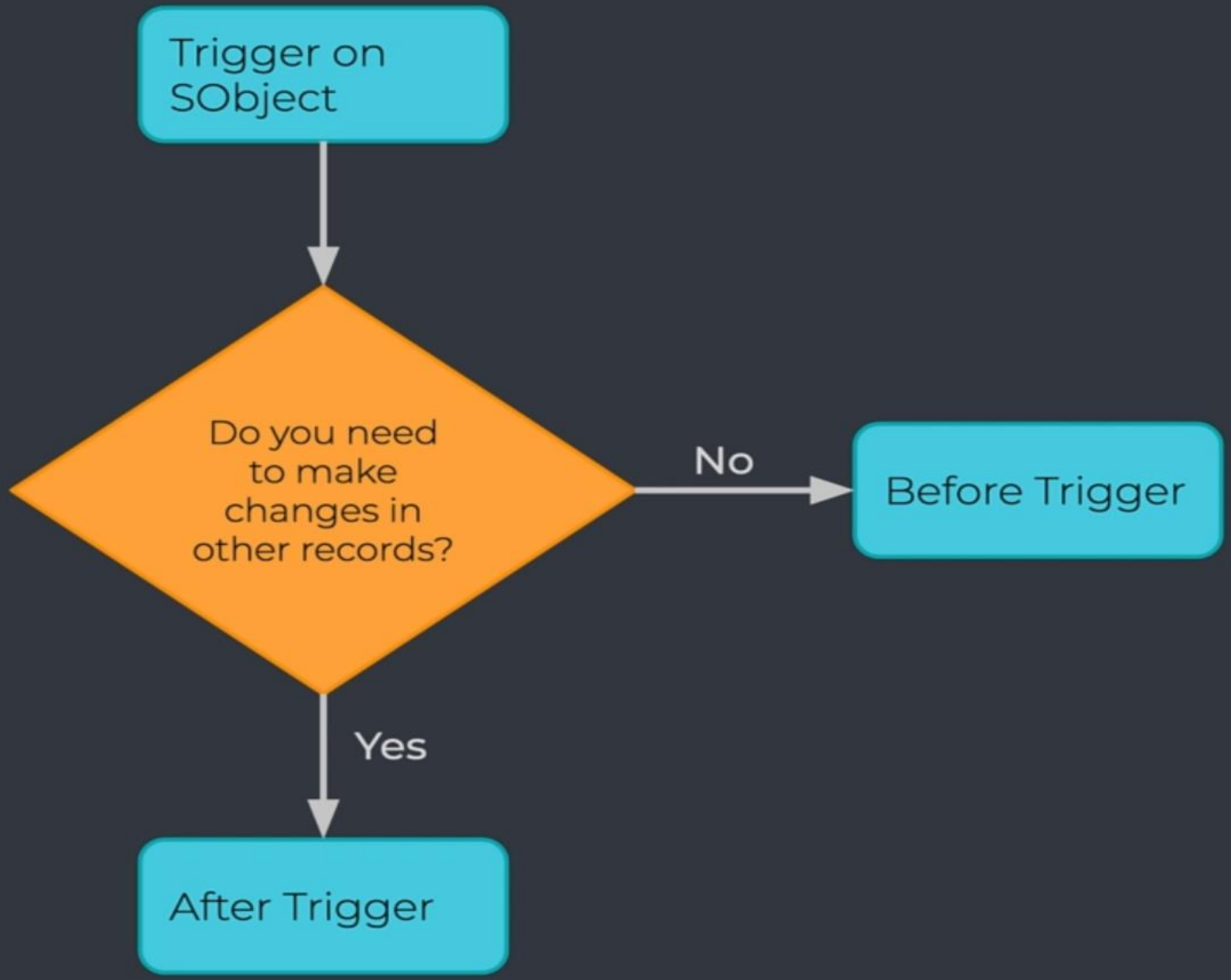
Before vs After

Before triggers are used to update or validate record values before they're saved to the database.

Avoid making changes to other records in before triggers.

Records that fire the after trigger, are read only.

Make changes (create/update/delete) to other records in after triggers.



Created BY DHARMESH GUPTA
8/29/2022

Order of Execution

Annotations

An Apex annotation modifies the way that a method/variable or class is used.

```
@deprecated
// This method is deprecated.
// Use playPubG2020() instead.
public void playPubG(){
    // Method to play pubg
    // This method does not support newer pubg versions
    System.debug('My old method to play pubg.')
}
```

```
public class AccountController {

    @AuraEnabled(cacheable=true)
    public static List<Account> getAllAccount() {
        return accounts;
    }
}
```

Annotation

Created BY DHARMESH GUPTA
8/29/2022

Annotations

```
@AuraEnabled
@Deprecated
@Future
@InvocableMethod
@InvocableVariable
@IsTest
@ReadOnly
@RemoteAction
@TestSetup
@TestVisible
```

Unit Testing

Test individual parts (units) of your software.

You do that by defining results that your code should produce for certain input values and verify whether they occur.

Created BY DHARMESH GUPTA
8/29/2022

Apex Testing

Testing is the key to successful long-term development and is a critical component of the development process.

To deploy your code in production org, you need at least 75% code coverage.

Test Runner

A unit test is a function that is executed by a software called a test-runner.

While writing tests, you define

- a short explanation on what you are testing
- a part of your software that you want to test (for ex. a function or a class)
- test data to supply as input or parameter to your code
- a result that you expect your code to produce

Assert Methods


```
System.assert(a == b);

System.assert(a == b, 'A is not equal to B');

System.assertEquals(a, b);

System.assertEquals(a, b, 'A is not equal to B');

System.assertNotEquals(a, b);

System.assertNotEquals(a, b, 'A is equal to B');
```

Created BY DHARMESH GUPTA
8/29/2022

Test Data

You need to create test data in order to test your organisation's apex classes and triggers.

Apex test data is transient and isn't committed to the database.

Test.startTest()

The startTest method marks the point in your test code when your test actually begins.

Each test method is allowed to call this method only once.

Test setup methods

Test setup methods enable you to create common test data easily and efficiently.

Test setup methods are defined in a test class, take no arguments, and return no value.

seeAllData=true

Annotate your test class or test method with IsTest(SeeAllData=true) to open up data access to records in your organization.

Advantages of Async Apex

Runs in background

Higher governor limits

Can be scheduled

Can process more number of records

Created BY DHARMESH GUPTA
8/29/2022

Async Features

Future Methods

Batch Apex

Queueable Apex

Scheduled Apex

Created BY DHARMESH GUPTA
8/29/2022

Future Method

```
global class MyClass {
    @future
    public static void myFutureMethod(){
        // Perform some operations
    }
}
```

- Runs in its own thread/transaction
- Executes when system resources are available

- Must be static
- Must have void return type
- Parameters must be primitive data types, collection of primitive data types

Batch Apex

You can use Batch Apex to build complex, long-running processes that run on thousands of records on the Lightning Platform.

- Operates over small batches/chunks of records.
- Can be scheduled
- Implement Database.Batchable interface

Batch Apex


```apex
public class SampleBatchClass implements Database.Batchable<Account>{
  public Database.QueryLocator start(Database.BatchableContext BC){
       // batch scope in this method
       String query = 'SELECT Id, Name FROM Account';
       return Database.getQueryLocator(query);
  }
  public void execute(Database.BatchableContext BC,
                      List<Account> scope){
       // process each chunk of records in this method
       for(Account acc : scope){
           acc.Rating = 'Hot';
       }
       update scope;
  }
  public void finish(Database.BatchableContext BC){
      // Do final processing in this method
      // like sending email or calling another batch class
  }
}
```

Executes only once

Return batch scope or records

Can return upto 50 million records

Executes for each batch/chunk of records

Must have void return type

Each execution will get new set of governor limits

Executes only once

Called after all batches are processed

Can be used for post processing

Batch Apex

```
SampleBatchClass batchInstance = new
SampleBatchClass();

// execute batch with batch size 50
Id batchJobId = Database.executeBatch(batchInstance,
50);

// schedule batch
String cronID = System.scheduleBatch(batchInstance, 'My
sample job', 30);
```

Param 1: Batch instance

Param 2: Optional scope parameter

Scope value must be greater than 0 and can be max 2000. Default value is 200.

Param 1: Batch instance

Param 2: Batch job name

Param 3 :Minutes from now

Queueable Apex

Queueable Apex is essentially a superset of future methods.

- Non-primitive types
- Monitoring
- Chaining jobs

Created BY DHARMESH GUPTA
8/29/2022

Queueable Apex


```
public class SampleQueueableClass implements Queueable {
    public void execute(QueueableContext context) {
        // write all aynchronous code here
        // you can chain other queueable class as well
    }
}
```

```
SampleQueueableClass sampleQue = new SampleQueueableClass();
// enqueue the job for processing
ID jobID = System.enqueueJob(sampleQue);
```

Scheduled Apex

Scheduled Apex provides you an ability to run you apex code at specific times or periodically.

Scheduled Apex


```
SampleScheduler sampleScheduler = new SampleScheduler();

String cronExp = '00 30 8 1 * ?';

String jobId = System.schedule('Merge Job', cronExp, sampleScheduler);
```

Created BY DHARMESH GUPTA
8/29/2022

Scheduled Apex

```
String cronExp = '00 30 8 1 * ?';
```

Name	Values	Special Chars
Seconds	0-59	
Minutes	0-59	
Hours	0-23	
Day of month	1-31	, - * ? / L W
Month	1-12 (JAN-DEC)	, - * /
Day of week	1-7 (SUN-SAT)	, - * ? / L #
Optional Year	null or 1970-2099	, - * /

, | Give multiple values > 1,2

- | Give range of values > 1-5

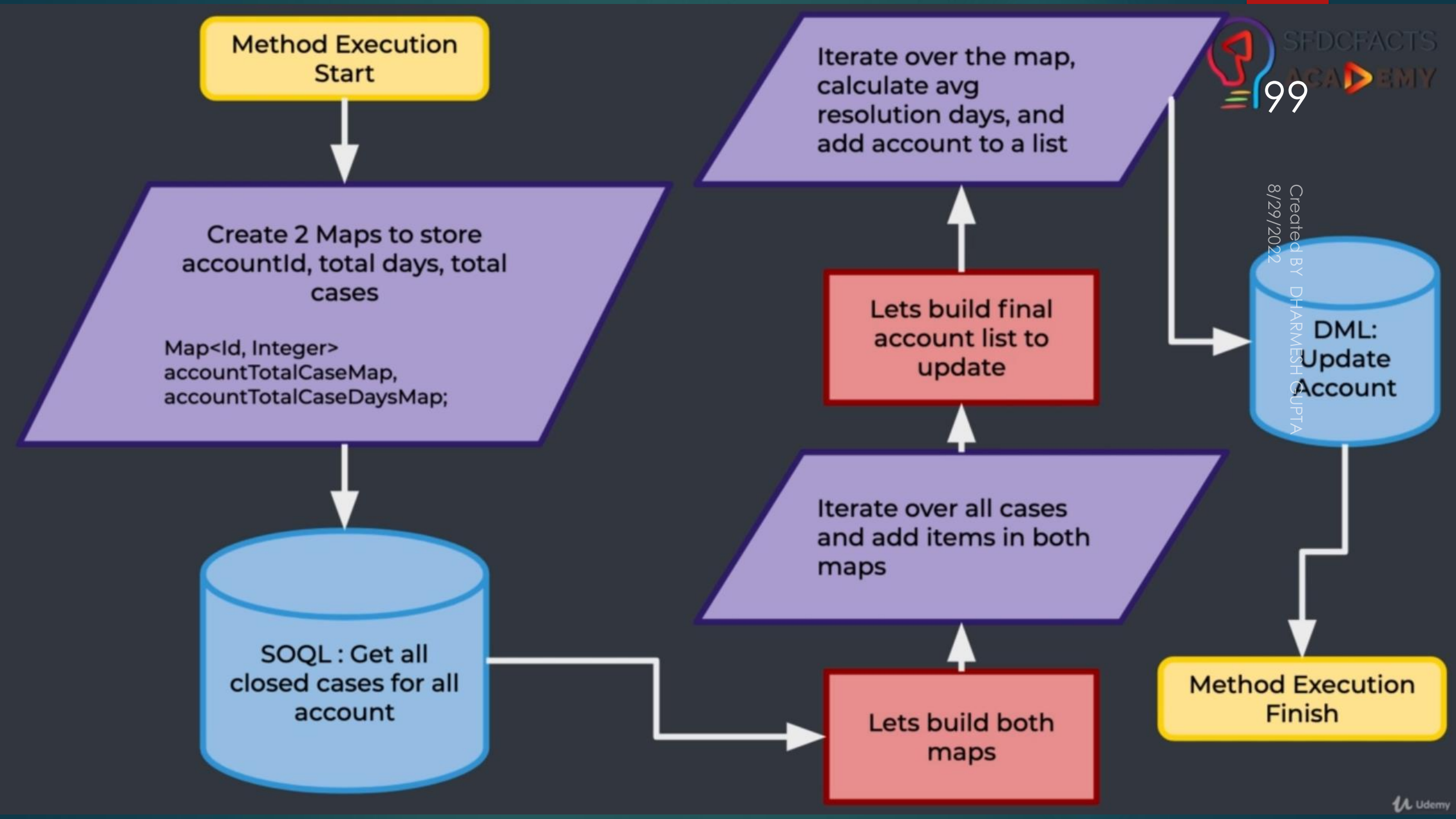
***** | All possible values

? | Do not specify any value

/ | Specifies increments

L | Last possible value

W | Closest weekday


Method Execution Start
Create 2 Maps to store accountId, total days, total cases
Map<Id, Integer> accountTotalCaseMap, accountTotalCaseDaysMap;
SOQL : Get all closed cases for all account
Lets build both maps
Iterate over all cases and add items in both maps
Lets build final account list to update
Iterate over the map, calculate avg resolution days, and add account to a list
DML: Update Account
Method Execution Finish
SFDCFACTS ACADEMY
99
Created BY DHARMESH GUPTA
8/29/2022
Udemy

Created BY DHARMESH GUPTA
8/29/2022

END