Q1:Create a content type named Employee with following fields:

Employee name

Employee Description

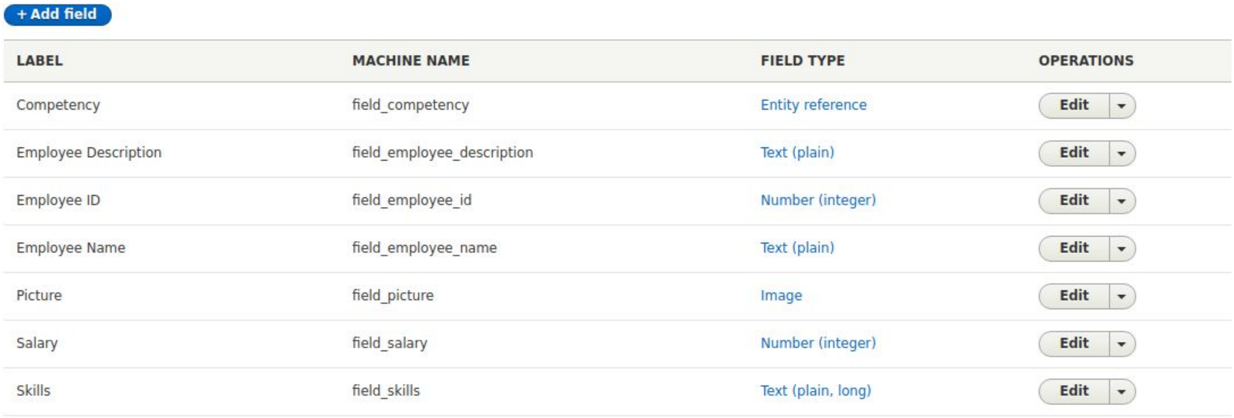
Employee ID

Picture

Salary

Skills

Competency(taxonomy term)



LABEL	MACHINE NAME	FIELD TYPE	OPERATIONS
Competency	field_competency	Entity reference	Edit ▾
Employee Description	field_employee_description	Text (plain)	Edit ▾
Employee ID	field_employee_id	Number (integer)	Edit ▾
Employee Name	field_employee_name	Text (plain)	Edit ▾
Picture	field_picture	Image	Edit ▾
Salary	field_salary	Number (integer)	Edit ▾
Skills	field_skills	Text (plain, long)	Edit ▾

Q2.Create a block programmatically listing all employee name and Employee ID.

Step 1:- under custom folder create a block let say the block myblock

step 2:- inside this folder create a file of myblock.info.yml and add the following code :-

```
name: Myblock

type: module

description: Defines a custom block.

core: 8.x

package: custom

dependencies:

- blocK
```

Step 3 :- create a folder named src

step 4:- create a subfolder named plugin that contains another subfolder named Block

Step 5:- Inside the folder Block create a file named myblock.php and add the following code :-

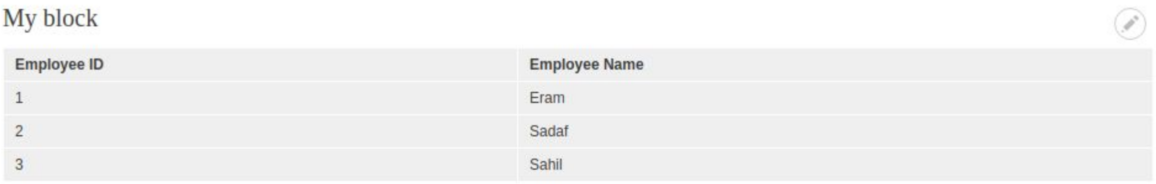
```
<?php


namespace Drupal\myblock\Plugin\Block;



use Drupal\Core\Access\AccessResult;

use Drupal\Core\Block\BlockBase;

use Drupal\Core\Form\FormStateInterface;

use Drupal\Core\Session\AccountInterface;



/**
```

```
* Provides a block with a simple text.
*
* @Block(
* id = "myblock_block",
* admin_label = @Translation("My block"),
* )
*/
class MyBlock extends BlockBase {
/**
* {@inheritdoc}
*/
public function build() {
$query = \Drupal::entityQuery('node')->condition('status', 1); //content type
$nids = $query->execute();
$url = '<table>';
$url .= "<tr><td><b>" . "Node ID" . '</td><td><b>' . "Node Name".
'</td></tr>';
foreach ($nids as $nid) {
$node = \Drupal\node\Entity\Node::load($nid);
$body = $node->body->value;
$title = $node->title->value;
$type =$node->type->value;
$url .= "<tr><td>" . $nid . '</td><td>' . $node->title->value . '</td></tr>';
}
$url .= "</table>";
```

```php
return [
'#markup' => $url,
];
}

/**
* {@inheritdoc}
*/
protected function blockAccess(AccountInterface $account) {
return AccessResult::allowedIfHasPermission($account, 'access content');
}

/**
* {@inheritdoc}
*/
public function blockForm($form, FormStateInterface $form_state) {
$config = $this->getConfiguration();

return $form;
}

/**
* {@inheritdoc}
*/
public function blockSubmit($form, FormStateInterface $form_state) {
```

```
$this->configuration['my_block_settings'] =
$form_state->getValue('my_block_settings');

}

}
```

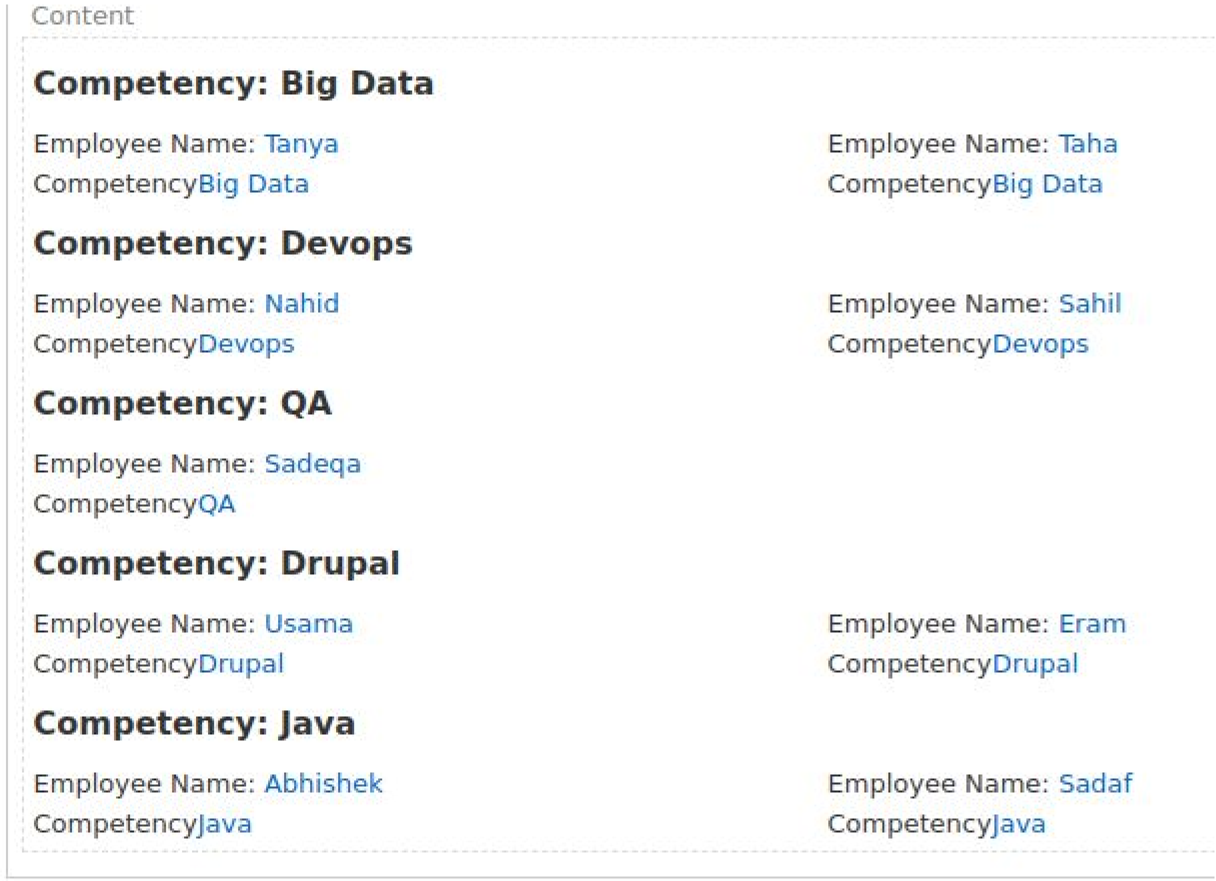
Step 6:- install the module and place your block at desired position and clear the cache



My block

Employee ID	Employee Name
1	Eram
2	Sadaf
3	Sahil

Q3:Create a view block of employee content type showing employee name and competency in grid format with 3 columns and group them on the basis of competency.



Content

Competency: Big Data

Employee Name: Tanya CompetencyBig Data	Employee Name: Taha CompetencyBig Data

Competency: Devops

Employee Name: Nahid CompetencyDevops	Employee Name: Sahil CompetencyDevops

Competency: QA

Employee Name: Sadeqa CompetencyQA

Competency: Drupal

Employee Name: Usama CompetencyDrupal	Employee Name: Eram CompetencyDrupal

Competency: Java

Employee Name: Abhishek CompetencyJava	Employee Name: Sadaf CompetencyJava

Q4.List the naming conventions for views templates for views created in question 5 and question 6.

All views templates can be overridden with a variety of names, using the view, the display ID of the view, the display type of the view, or some combination thereof.

For each view, there will be a minimum of two templates used. The first is used for all **views: views-view.html.twig.**

The second template is determined by the style selected for the view. Note that certain aspects of the view can also change which style is used; for example, arguments which provide a summary view might change the style to one of the special summary styles.

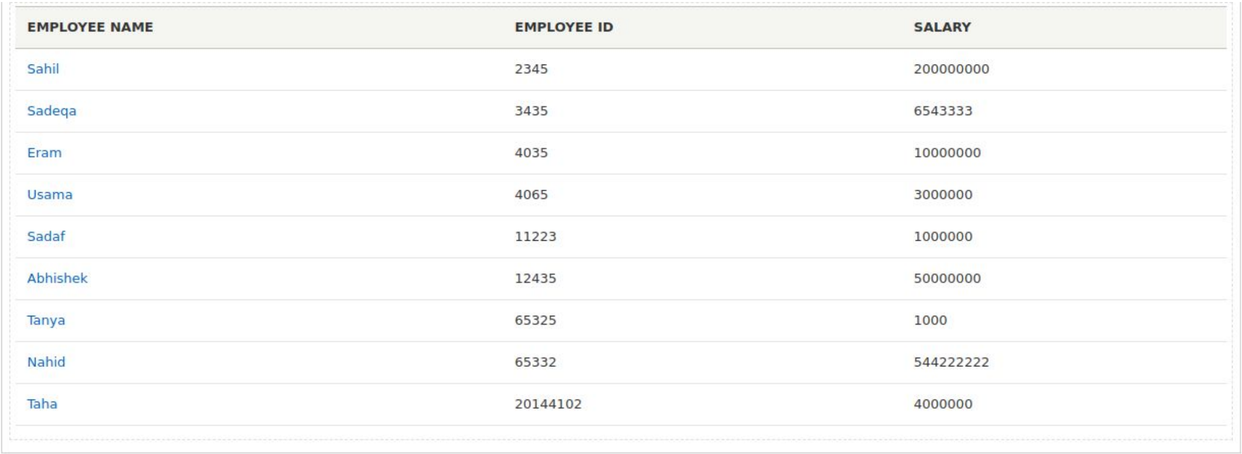
The default style for all views is **views-view-unformatted.html.twig.**

Many styles will then farm out the actual display of each row to a row style; the default row style is **views-view-fields.html.twig.**

Q5:Create a view page of employee content type in a table format with Employee name,

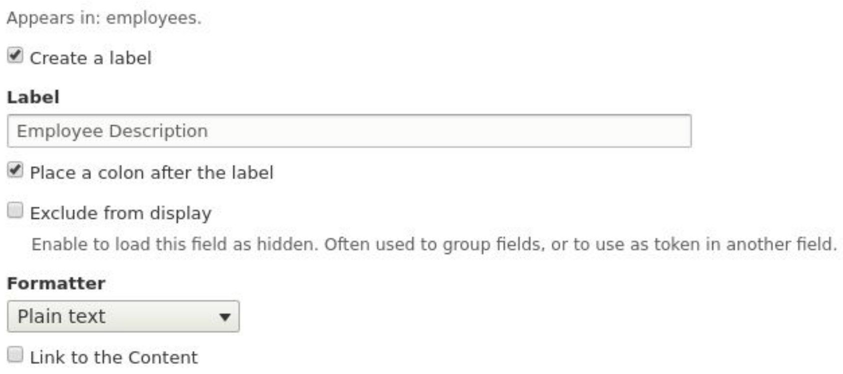
Employee ID and salary as header

1. Sort it on the basis of Employee ID



EMPLOYEE NAME	EMPLOYEE ID	SALARY
Sahil	2345	200000000
Sadeqa	3435	6543333
Eram	4035	10000000
Usama	4065	3000000
Sadaf	11223	1000000
Abhishek	12435	50000000
Tanya	65325	1000
Nahid	65332	544222222
Taha	20144102	4000000

2. Trim the Employee Description field to 50 characters.



Appears in: employees.

☑ Create a label

Label

Employee Description

☑ Place a colon after the label

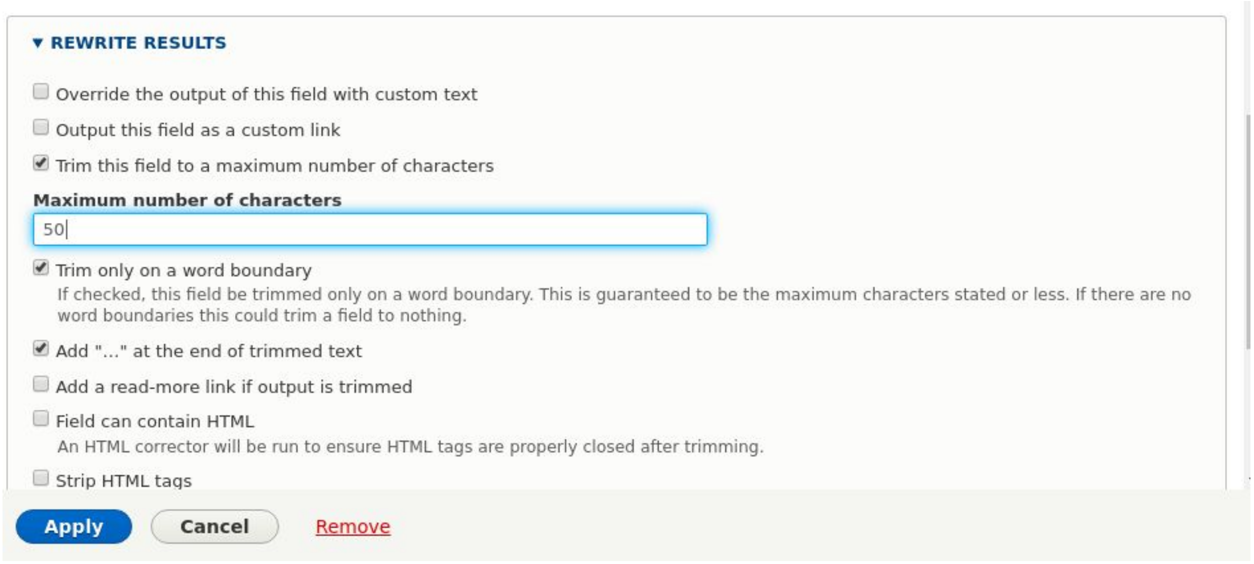
□ Exclude from display

Enable to load this field as hidden. Often used to group fields, or to use as token in another field.

Formatter

Plain text ▾

□ Link to the Content



▾ REWRITE RESULTS

□ Override the output of this field with custom text

□ Output this field as a custom link

☑ Trim this field to a maximum number of characters

Maximum number of characters

50

☑ Trim only on a word boundary

If checked, this field be trimmed only on a word boundary. This is guaranteed to be the maximum characters stated or less. If there are no word boundaries this could trim a field to nothing.

☑ Add "..." at the end of trimmed text

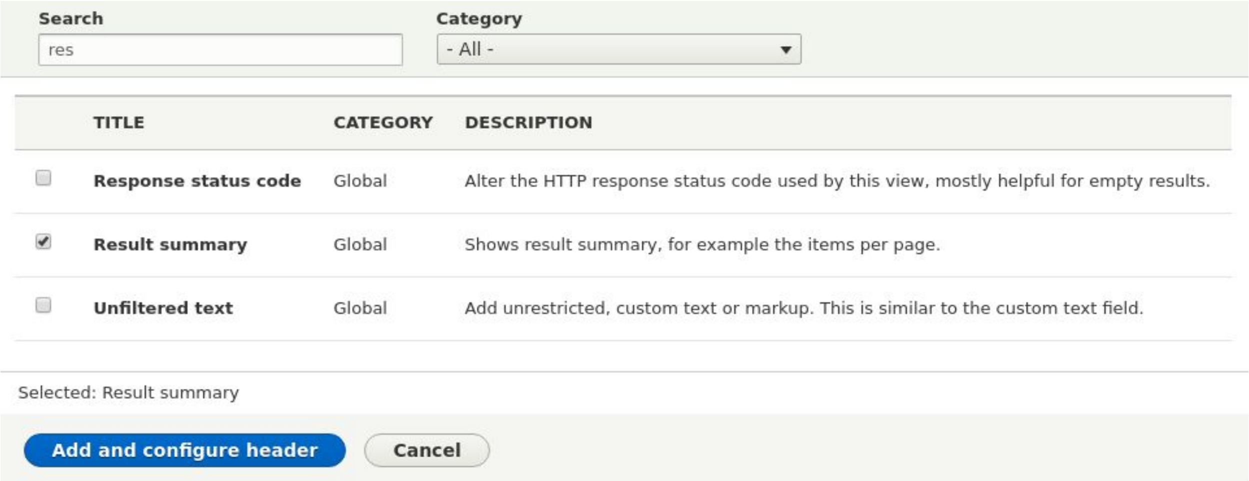
□ Add a read-more link if output is trimmed

□ Field can contain HTML

An HTML corrector will be run to ensure HTML tags are properly closed after trimming.

□ Strip HTML tags

3. Add header “Total Records found : total”

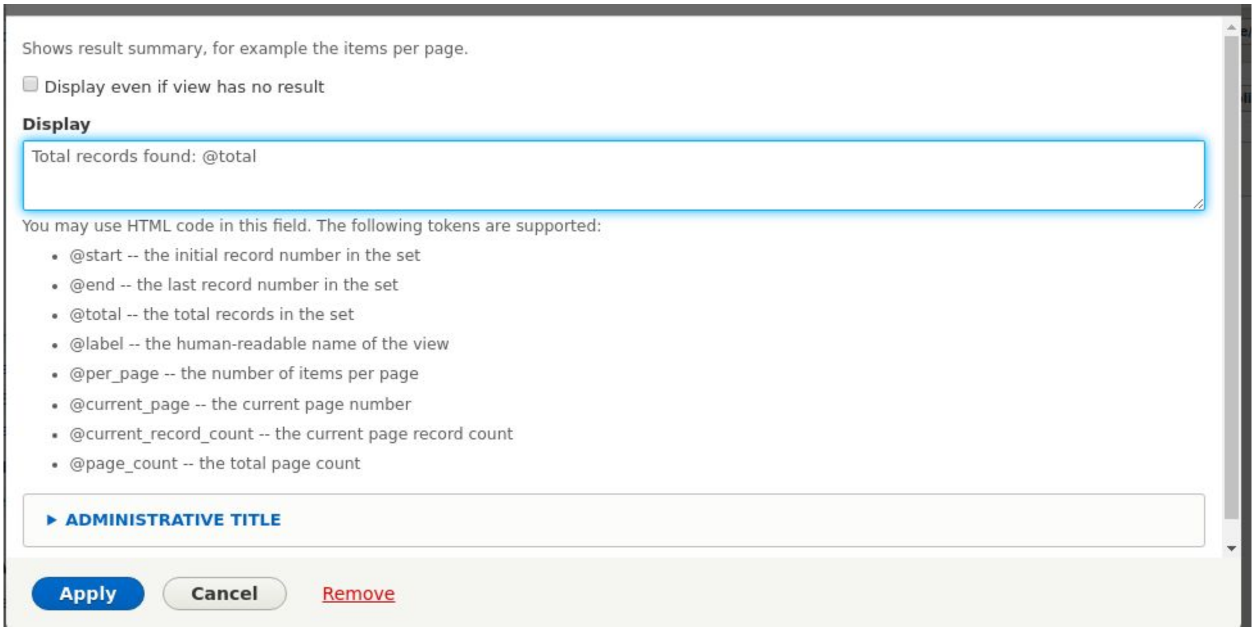


Search: res

Category: - All -

	TITLE	CATEGORY	DESCRIPTION
☐	**Response status code**	Global	Alter the HTTP response status code used by this view, mostly helpful for empty results.
☑	**Result summary**	Global	Shows result summary, for example the items per page.
☐	**Unfiltered text**	Global	Add unrestricted, custom text or markup. This is similar to the custom text field.

Selected: Result summary



Shows result summary, for example the items per page.

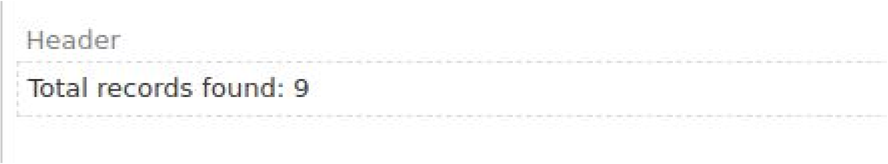
☐ Display even if view has no result

Display

Total records found: @total

You may use HTML code in this field. The following tokens are supported:

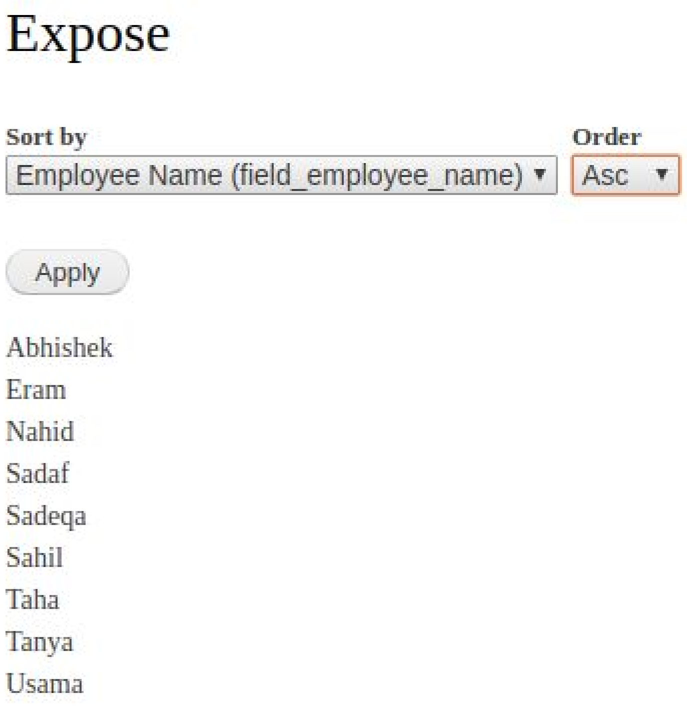
- @start -- the initial record number in the set
- @end -- the last record number in the set
- @total -- the total records in the set
- @label -- the human-readable name of the view
- @per_page -- the number of items per page
- @current_page -- the current page number
- @current_record_count -- the current page record count
- @page_count -- the total page count



Header

Total records found: 9

4. Expose the filter for employee name



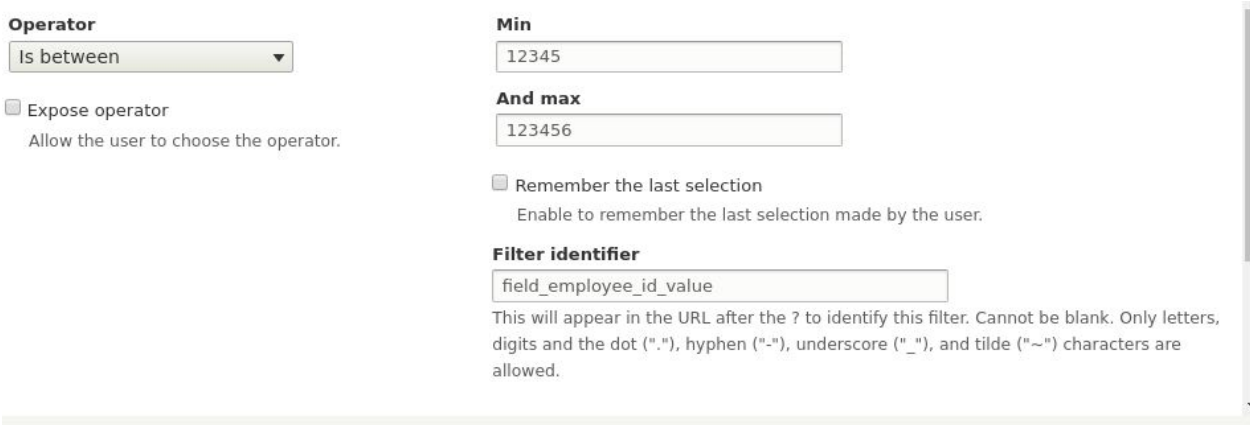
Expose

Sort by
Employee Name (field_employee_name) ▾

Order
Asc ▾

Abhishek
Eram
Nahid
Sadaf
Sadeqa
Sahil
Taha
Tanya
Usama

5. Expose the filter for employee id (use Between)



Operator

Is between ▾

□ Expose operator

Allow the user to choose the operator.

Min

12345

And max

123456

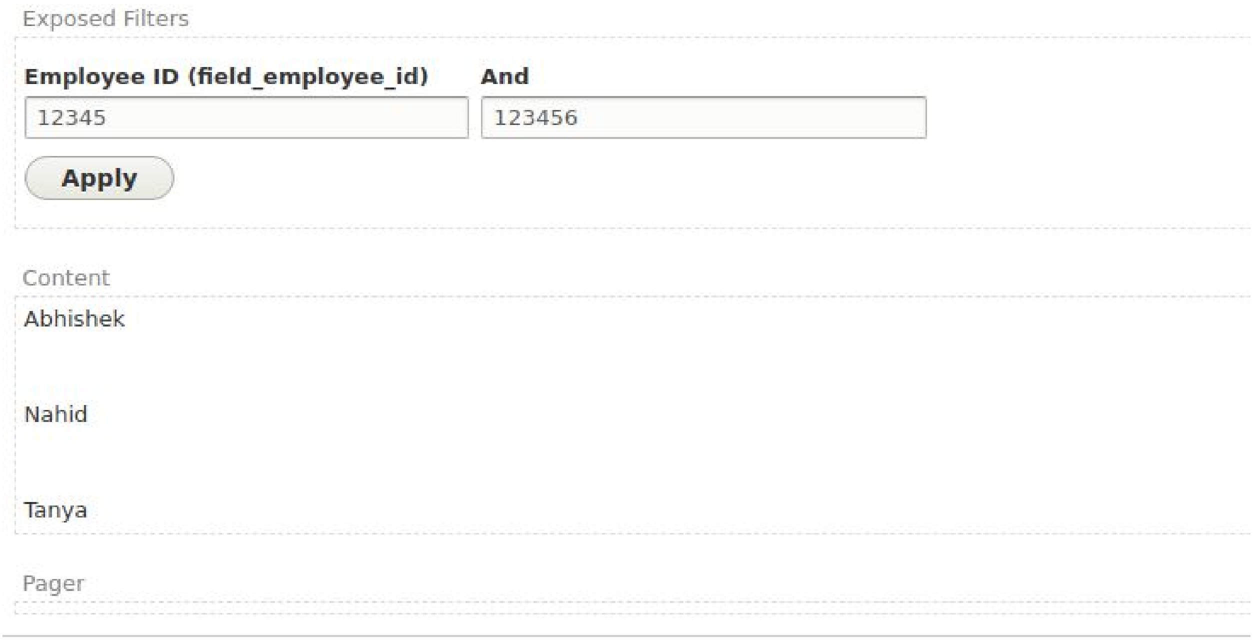
□ Remember the last selection

Enable to remember the last selection made by the user.

Filter identifier

field_employee_id_value

This will appear in the URL after the ? to identify this filter. Cannot be blank. Only letters, digits and the dot ("."), hyphen ("-"), underscore ("_"), and tilde ("~") characters are allowed.



Exposed Filters

Employee ID (field_employee_id) 12345 **And** 123456

Content

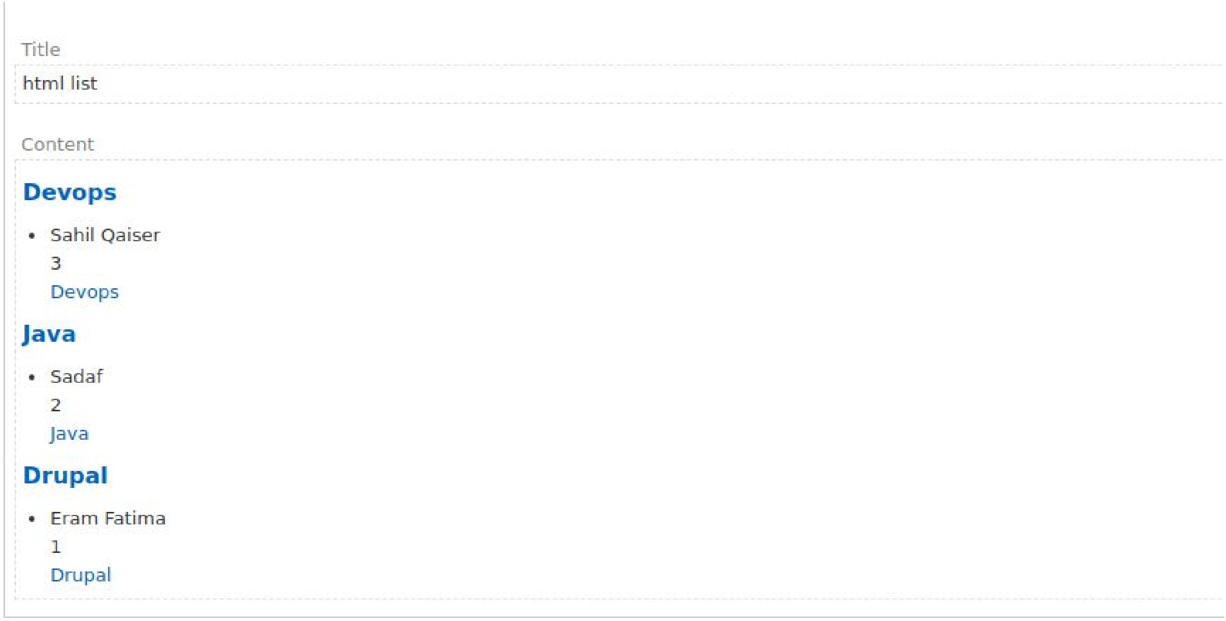
Abhishek

Nahid

Tanya

Pager

Q6:Create a view to show the Employee details(name,id) in HTML List on the basis of competency(taxonomy term id)



Title

html list

Content

Devops

- Sahil Qaiser
  3
  Devops

Java

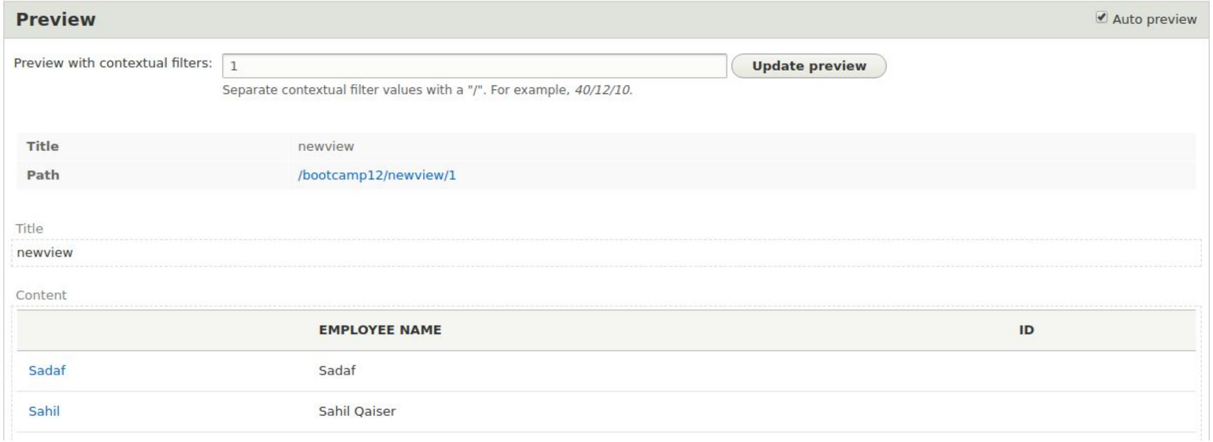
- Sadaf
  2
  Java

Drupal

- Eram Fatima
  1
  Drupal

Q7. Create a view listing Employee name based on node id and show it on employee content type.

Note: The title(employee name) should be excluded in which the view is placed.



Preview with contextual filters: 1

Separate contextual filter values with a "/". For example, *40/12/10.*

Title	newview
Path	/bootcamp12/newview/1

Title

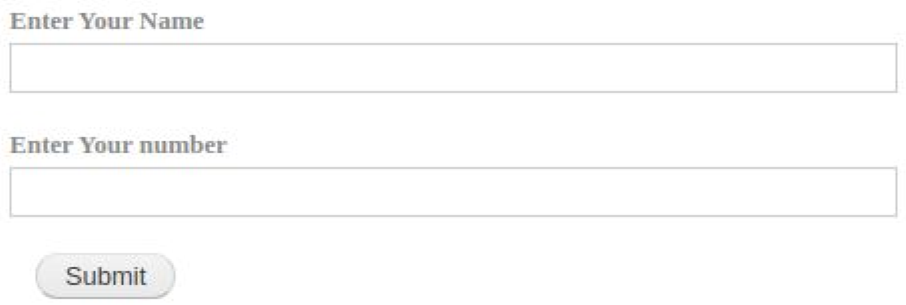
newview

Content

	EMPLOYEE NAME	ID
Sadaf	Sadaf	
Sahil	Sahil Qaiser	

Q8. Create a form with name and mobile fields and on submit show the field values entered inside a block programmatically.

Note: Last four question carry 3 marks and rest are of 2 marks each


Enter Your Name
Enter Your number
Submit