Account SCIM v2.1 [Public Preview]

Account SCIM v2.1 is a more scalable version of SCIM 2.0. It is the recommended method for syncing users and groups into Databricks.

Account Users

User identities recognized by Databricks and represented by email addresses.

Databricks recommends using SCIM provisioning to sync users and groups automatically from your identity provider to your Databricks account. SCIM streamlines onboarding a new employee or team by using your identity provider to create users and groups in a Databricks account and give them the proper level of access. When a user leaves your organization or no longer needs access to Databricks account, admins can terminate the user in your identity provider and that user's account will also be removed from Databricks account. This ensures a consistent offboarding process and prevents unauthorized users from accessing sensitive data.

GET List users

```
/api/2.1/accounts/{account_id}/scim/v2/Users
```

Gets details for all the users associated with an account. Will return at most 100 users at a time.

Query Params

filter	Query by which the results have to be filtered. Only exact filtering by userName is supported. Example: filter=username eq "user@example.com"
attributes	Comma-separated list of attributes to return in response.
excludedAttributes	Comma-separated list of attributes to exclude in response.
startIndex	Specifies the index of the first result. First item is number 1.
count	Desired number of results per page. Must be <= 100.

Example Response (200)


```
{
    "schemas": [
        "urn:ietf:params:scim:api:messages:2.0:ListResponse"
    ],
    "totalResults": 1,
    "itemsPerPage": 1,
    "startIndex": 1,
    "Resources": [
        {
            "schemas": [
                "urn:ietf:params:scim:schemas:core:2.0:User"
            ],
            "id": "string",
            "meta": {
                "resourceType": "User",
                "location": "string"
            },
            "userName": "user@example.com",
            "name": {
                "familyName": "string",
                "givenName": "string"
            },
            "displayName": "string",
            "active": true,
            "emails": [
                {
                    "type": "string",
                    "value": "user@example.com",
                    "primary": true
                }
            ]
        }
    ]
}
```


This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:


```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
```

```
}
```

GET Get user details

```
/api/2.1/accounts/{account_id}/scim/v2/Users/{id}
```

Gets the information for a specific user in the account.

Example Response (200)

```
{
    "emails": [
        {
            "type": "string",
            "value": "user@example.com",
            "primary": true
        }
    ],
    "displayName": "string",
    "name": {
        "familyName": "string",
        "givenName": "string"
    },
    "active": true,
    "userName": "user@example.com"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

POST Create a new user

```
/api/2.1/accounts/{account_id}/scim/v2/Users
```

Creates a new user in the account.

Body (json)

```
{
    "emails": [
        {
            "type": "string",
            "value": "user@example.com",
            "primary": true
        }
    ],
    "displayName": "string",
    "name": {
        "familyName": "string",
        "givenName": "string"
    },
    "active": true,
    "userName": "user@example.com"
}
```

Example Response (200)

```
{
    "emails": [
        {
            "type": "work",
            "value": "user@example.com",
            "primary": true
        }
    ],
    "displayName": "John Doe",
    "name": {
        "familyName": "Doe",
        "givenName": "John"
    },
    "active": true,
```

```
    "id": "string",
    "userName": "user@example.com"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 409, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

PATCH Update user details

```
/api/2.1/accounts/{account_id}/scim/v2/Users/{id}
```

Partially updates the details of a user.

Body (json)

Body (json)

Add roles

```
{
   "schemas": [
       "urn:ietf:params:scim:api:messages:2.0:PatchOp"
   ],
   "Operations": [
       {
           "op": "add",
           "path": "roles",
           "value": [
               {
                   "value": "account_admin"
               }
           ]
       }
   ]
```

```
}
```

Remove roles

```
{
   "schemas": [
       "urn:ietf:params:scim:api:messages:2.0:PatchOp"
   ],
   "Operations": [
       {
           "op": "remove",
           "path": "roles[value eq \"account_admin\"]"
       }
   ]
}
```

Other modifiable attributes include displayName and active.

Example Response (204)

```
{
    "emails": [
        {
            "type": "string",
            "value": "user@example.com",
            "primary": true
        }
    ],
    "displayName": "string",
    "name": {
        "familyName": "string",
        "givenName": "string"
    },
    "active": true,
    "userName": "user@example.com"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
```

```
  "message": "Human-readable error message."
}
```

PUT Replace a user

```
/api/2.1/accounts/{account_id}/scim/v2/Users/{id}
```

Replaces a user's information with the data supplied in request.

Body (json)

```
{
    "emails": [
        {
            "type": "string",
            "value": "user@example.com",
            "primary": true
        }
    ],
    "displayName": "string",
    "name": {
        "familyName": "string",
        "givenName": "string"
    },
    "active": true,
    "userName": "user@example.com"
}
```

Example Response (200)

```
{
    "emails": [
        {
            "type": "string",
            "value": "user@example.com",
            "primary": true
        }
    ],
```

```
    "displayName": "string",
    "name": {
        "familyName": "string",
        "givenName": "string"
    },
    "active": true,
    "userName": "user@example.com"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

DELETE Delete a user

```
/api/2.1/accounts/{account_id}/scim/v2/Users/{id}
```

Deletes a user from the account.

Example Response (204)

```
No content
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

Account Groups

Groups simplify identity management, making it easier to assign access to Azure Databricks account, data, and other securable objects.

It is best practice to assign access to workspaces and access-control policies in Unity Catalog to groups, instead of to users individually. All Azure Databricks account identities can be assigned as members of groups, and members inherit permissions that are assigned to their group.

GET List groups

```
/api/2.1/accounts/{account_id}/scim/v2/Groups
```

Gets all details of the groups associated with the account. Will return at most 100 groups at a time.

Query Params

filter	Query by which the results have to be filtered. Only exact filtering by displayName or externalId is supported Examples: filter=displayName eq "foo" filter=externalId eq "bar"
attributes	Comma-separated list of attributes to return in response.
excludedAttributes	Comma-separated list of attributes to exclude in response.
startIndex	Specifies the index of the first result. First item is number 1.
count	Desired number of results per page. Must be <= 100.

Example Response (200)

```
{
    "totalResults": 1,
    "startIndex": 1,
    "itemsPerPage": 1,
    "Resources": [
```

```
        {
            "id": "string",
            "displayName": "string",
            "roles": [
                {
                    "value": "string"
                }
            ],
            "externalId": "string"
        }
    ]
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

GET Get group details

```
/api/2.1/accounts/{account_id}/scim/v2/Groups/{id}
```

Gets the information for a specific group in the account.

Example Response (200)

```
{
    "id": "12345678",
    "displayName": "string",
    "members": [
        {
            "$ref": "string",
            "value": "string",
            "display": "string",
        }
    ],
    "roles": [
```

```
        {
            "$ref": "string",
            "value": "string",
            "display": "string"
        }
    ],
    "externalId": "string"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

POST Create a new group

```
/api/2.1/accounts/{account_id}/scim/v2/Groups
```

Creates a group in the account with a unique name, using the supplied group details. It's recommended to create groups initially, then add group memberships through PATCH. The maximum number of members a group can be created with is 5000.

Body (json)

```
{
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:Group"
    ],
    "displayName": "string",
    "members": [
        {
            "value": "string"
        }
    ]
}
```

Example Response (200)

```
{
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:Group"
    ],
    "displayName": "string",
    "members": [
        {
            "$ref": "string",
            "value": "string",
            "display": "string"
        }
    ],
    "id": "759734401230411"
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 409, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

PATCH Update group details

```
/api/2.1/accounts/{account_id}/scim/v2/Groups/{id}
```

Partially updates the details of a group.

Body (json)

Add member to group

```
{
   "schemas": [
       "urn:ietf:params:scim:api:messages:2.0:PatchOp"
   ],
   "Operations": [
```

```
        {
            "op": "add",
            "value": {
                "members": [
                    {
                        "value": "<userId>"
                    }
                ]
            }
        }
    ]
}
```

Remove member from group

```
{
    "schemas": [
        "urn:ietf:params:scim:api:messages:2.0:PatchOp"
    ],
    "Operations": [
        {
            "op": "remove",
            "path": "members[value eq \"<userId>\"]"
        }
    ]
}
```

Other modifiable attributes include displayName, and roles.

Example Response (204)

```
No content
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

DELETE Delete a group

```
/api/2.1/accounts/{account_id}/scim/v2/Groups/{id}
```

Deletes a group from the account.

Example Response (204)

```
No content
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

Account Service Principals

Identities for use with jobs, automated tools, and systems such as scripts, apps, and CI/CD platforms. Databricks recommends creating service principals to run production jobs or modify production data. If all processes that act on production data run with service principals, interactive users do not need any write, delete, or modify privileges in production. This eliminates the risk of a user overwriting production data by accident.

GET List service principals

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals
```

Gets the set of service principals associated with an account. Will return at most 100 service principals at a time.

Query Params

filter	Query by which the results have to be filtered. Only exact filtering by applicationId is supported. Example: filter=applicationId eq "6ef14233-a641-4f1e-905a-9158cbd3b353"
attributes	Comma-separated list of attributes to return in response.
excludedAttributes	Comma-separated list of attributes to exclude in response.
startIndex	Specifies the index of the first result. First item is number 1.
count	Desired number of results per page. Must be <= 100.

Example Response (200)


```
{
    "schemas": [
        "urn:ietf:params:scim:api:messages:2.0:ListResponse"
    ],
    "totalResults": 1,
    "itemsPerPage": 1,
    "startIndex": 1,
    "Resources": [
        {
            "schemas": [
                "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
            ],
            "id": "string",
            "meta": {
                "resourceType": "ServicePrincipal",
                "location": "string"
            },
            "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
            "displayName": "string",
            "active": true
        }
    ]
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

GET Get service principal details

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals/{id}
```

Gets the information for a specific service principal in the account.

Example Response (200)

```
{
    "displayName": "string",
    "id": "string",
    "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
    ],
    "active": true
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

POST Create a new service principal

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals
```

Creates a new service principal in the account.

Body (json)

```
{
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:User"
    ],
    "displayName": "etl-service"
}
```

Example Response (200)

```
{
    "displayName": "etl-service",
    "id": "string",
    "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
    ],
    "active": true
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 409, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

PATCH Update service principal details

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals/{id}
```

Partially updates the details of a service principal.

Body (json)

Add roles

```
{
   "schemas": [
       "urn:ietf:params:scim:api:messages:2.0:PatchOp"
   ],
   "Operations": [
       {
           "op": "add",
           "path": "roles",
           "value": [
               {
                   "value": "account_admin"
               }
           ]
       }
   ]
}
```

Remove roles

```
{
   "schemas": [
       "urn:ietf:params:scim:api:messages:2.0:PatchOp"
   ],
   "Operations": [
       {
           "op": "remove",
           "path": "roles[value eq \"account_admin\"]"
       }
   ]
}
```

Other modifiable attributes include displayName and active.

Example Response (204)

```
{
    "displayName": "string",
    "id": "string",
```

```
    "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
    ],
    "active": true
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

PUT Replace a service principal

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals/{id}
```

Replaces a service principal's information with the data supplied in request.

Body (json)

```
{
    "displayName": "string",
    "id": "string",
    "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
    ],
    "active": true
}
```

Example Response (204)

```
{
    "displayName": "string",
    "id": "string",
```

```
    "applicationId": "6ef14233-a641-4f1e-905a-9158cbd3b353",
    "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
    ],
    "active": true
}
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```

DELETE Delete a service principal

```
/api/2.1/accounts/{account_id}/scim/v2/ServicePrincipals/{id}
```

Deletes a service principal from the account.

Example Response (204)

```
No content
```

This might return the following HTTP codes: 400, 401, 403, 404, 429, 500. Error responses are returned in the following format:

```
{
  "error_code": "Error code",
  "message": "Human-readable error message."
}
```