

Contabilidad

Bienvenido al **libro contable digital** de tu inmobiliaria. En este apartado centralizamos las herramientas de conexión para que tus plataformas externas consulten y sincronicen tu información financiera de forma automática.

¿Para qué sirve este módulo? El objetivo principal es eliminar los procesos manuales en la extracción de datos financieros. En lugar de descargar informes a mano, tus sistemas podrán consultar directamente el historial y los movimientos de tus cuentas contables. Podrás utilizar filtros estratégicos —como rangos de fechas o grupos de cuentas— para extraer exactamente la información que necesitas sin sobrecargar tus integraciones.

Explora el listado a continuación para descubrir cómo alimentar tus tableros de análisis y mantener tus números perfectamente organizados, auditables y sincronizados.

- [Reporte Auxiliar General](#)

Reporte Auxiliar General

Permite obtener el reporte contable **Auxiliar General**, que presenta los movimientos detallados de cada cuenta del Plan Único de Cuentas (PUC) dentro de un período y rango de cuentas. Cada elemento incluye el saldo anterior, débitos, créditos, saldo actual, base de retención, y el desglose por tercero con sus respectivos documentos contables.

¿Para qué sirve este servicio?

Úsalo para alimentar reportes contables en tu ERP, conciliar saldos con sistemas externos, generar informes de auditoría, exportar movimientos detallados a Excel o Power BI, o validar la contabilidad de un período específico.

1. El Endpoint (La dirección web)

Apunta tu sistema a la siguiente dirección. Recuerda reemplazar **{{instancia}}** por la dirección web completa que utilizas para ingresar a tu plataforma.

```
GET https://{{instancia}}/service/v2/public/accounting/general-ledger
```

¿Qué debes colocar en {{instancia}}?

Es muy sencillo: corresponde a la **dirección web principal** que utilizas a diario para ingresar a tu plataforma (incluyendo la terminación `.nuby.app` o `.arrendasoft.co`).

Por ejemplo, si para entrar a tu sistema escribes `inmobiliaria.nuby.app` o `inmobiliaria.arrendasoft.co` en tu navegador, esa será exactamente tu instancia. Solo asegúrate de no incluir el "https://" ni barras diagonales ("/") al final.

2. La Petición (¿Qué debes enviarnos?)

Este servicio no requiere cuerpo en la petición. Envía los encabezados requeridos y opcionalmente los parámetros de filtro en la URL:

Método	GET
Content-Type	application/json

Authorization	Bearer token , Token obtenido al consumir el servicio Login
----------------------	---

¿Aún no tienes tu Token de acceso?

Para consumir este servicio necesitas un Token vigente. Consulta el servicio de [Login](#) para aprender cómo obtenerlo.

Parámetros de consulta (query string):

Parámetro	Tipo	Requerido	Por defecto	Descripción
<code>fecha_ini</code>	string	No	Primer día del año en curso	Fecha inicial del rango a consultar. Formato: <code>YYYY-MM-DD</code> .
<code>fecha_fin</code>	string	No	Último día del mes en curso	Fecha final del rango a consultar. Formato: <code>YYYY-MM-DD</code> . Debe ser mayor o igual a <code>fecha_ini</code> .
<code>cuenta_ini</code>	string	No	1	Código inicial de cuenta PUC. Solo permite valores numéricos. Debe ser mayor o igual a <code>1</code> .
<code>cuenta_fin</code>	string	No	9	Código final de cuenta PUC. Solo permite valores numéricos. Debe ser mayor o igual a <code>cuenta_ini</code> .

Ejemplos de petición:

```
GET https://mi-inmobiliaria.nuby.app/service/v2/public/accounting/general-ledger
GET https://mi-inmobiliaria.nuby.app/service/v2/public/accounting/general-ledger?fecha_ini=2025-01-01
GET https://mi-inmobiliaria.nuby.app/service/v2/public/accounting/general-ledger?fecha_ini=2025-01-01&fecha_fin=2025-06-30
GET https://mi-inmobiliaria.nuby.app/service/v2/public/accounting/general-ledger?fecha_ini=2025-01-01&fecha_fin=2025-06-30&cuenta_ini=1
GET https://mi-inmobiliaria.nuby.app/service/v2/public/accounting/general-ledger?fecha_ini=2025-01-01&fecha_fin=2025-06-30&cuenta_ini=1&cuenta_fin=4
```

3. La Respuesta (¿Qué te entregaremos?)

Si tu Token es válido y los parámetros son correctos, el sistema te devolverá un objeto JSON con el listado de cuentas contables, sus terceros y los movimientos detallados de cada uno. La respuesta tiene una estructura jerárquica de tres niveles: **Cuenta → Tercero → Detalle**.

```
{
  "error": false,
  "message": "",
  "body": [
    {
      "cuenta": "11050501",
      "nombre_cuenta": "Caja general moneda nacional",
      "saldo_anterior": 8750320.45,
      "debitos": 12480000,
      "creditos": 9230000,
      "saldo_actual": 12000320.45,
      "base_retencion": 0,
      "terceros": {
        "42": {
          "documento_tercero": "80213654",
          "nombre_tercero": "CARLOS EDUARDO PINEDA VARGAS",
          "saldo_anterior": 3250120.45,
          "debitos": 7500000,
          "creditos": 4230000,
          "saldo_actual": 6520120.45,
          "base_retencion": 0,
          "detalles": [
            {
              "fecha_documento": "2025-02-05",
              "consecutivo_documento": "RC-10450",
              "detalle_documento": "Pago canon arrendamiento febrero Apto 501",
              "debitos": 2850000,
              "creditos": 0,
              "base_retencion": 0
            },
            {
              "fecha_documento": "2025-03-03",
              "consecutivo_documento": "RC-10512",
              "detalle_documento": "Pago canon arrendamiento marzo Apto 501",
              "debitos": 2850000,
```

```
        "creditos": 0,
        "base_retencion": 0
    },
    {
        "fecha_documento": "2025-03-15",
        "consecutivo_documento": "EG-4021",
        "detalle_documento": "Consignación propietario marzo",
        "debitos": 0,
        "creditos": 4230000,
        "base_retencion": 0
    }
]
},
"73": {
    "documento_tercero": "1098745231",
    "nombre_tercero": "JULIANA PATRICIA RÍOS MENDOZA",
    "saldo_anterior": 1500200,
    "debitos": 4980000,
    "creditos": 5000000,
    "saldo_actual": 1480200,
    "base_retencion": 0,
    "detalles": [
        {
            "fecha_documento": "2025-01-10",
            "consecutivo_documento": "RC-10389",
            "detalle_documento": "Pago canon arrendamiento enero Local 3",
            "debitos": 4500000,
            "creditos": 0,
            "base_retencion": 0
        },
        {
            "fecha_documento": "2025-01-20",
            "consecutivo_documento": "EG-3998",
            "detalle_documento": "Consignación propietario enero",
            "debitos": 0,
            "creditos": 5000000,
            "base_retencion": 0
        }
    ]
}
```

```
    }
  },
  {
    "cuenta": "13050501",
    "nombre_cuenta": "Arrendamientos por cobrar",
    "saldo_anterior": 15400000,
    "debitos": 9350000,
    "creditos": 12480000,
    "saldo_actual": 12270000,
    "base_retencion": 0,
    "terceros": {
      "55": {
        "documento_tercero": "43876123",
        "nombre_tercero": "DIANA MARCELA OSPINA VELÁSQUEZ",
        "saldo_anterior": 2400000,
        "debitos": 3600000,
        "creditos": 4800000,
        "saldo_actual": 1200000,
        "base_retencion": 0,
        "detalles": [
          {
            "fecha_documento": "2025-01-01",
            "consecutivo_documento": "FV-8901",
            "detalle_documento": "Factura arrendamiento enero Cll 23 # 56 -
78",
            "debitos": 1200000,
            "creditos": 0,
            "base_retencion": 0
          },
          {
            "fecha_documento": "2025-01-15",
            "consecutivo_documento": "RC-10401",
            "detalle_documento": "Recibo de caja pago enero",
            "debitos": 0,
            "creditos": 1200000,
            "base_retencion": 0
          },
          {
            "fecha_documento": "2025-02-01",
            "consecutivo_documento": "FV-8950",
```

78",

```
        "detalle_documento": "Factura arrendamiento febrero Cll 23 # 56 -  
    },  
    "debitos": 1200000,  
    "creditos": 0,  
    "base_retencion": 0  
  },  
  {  
    "fecha_documento": "2025-02-10",  
    "consecutivo_documento": "RC-10465",  
    "detalle_documento": "Recibo de caja pago febrero",  
    "debitos": 0,  
    "creditos": 1200000,  
    "base_retencion": 0  
  }  
]  
}  
},  
{  
  "cuenta": "24080501",  
  "nombre_cuenta": "Retención en la fuente arrendamientos 3.5%",  
  "saldo_anterior": -875000,  
  "debitos": 0,  
  "creditos": 249375,  
  "saldo_actual": -1124375,  
  "base_retencion": 7125000,  
  "terceros": {  
    "65": {  
      "documento_tercero": "900456789",  
      "nombre_tercero": "INVERSIONES HORIZONTE S.A.S.",  
      "saldo_anterior": -875000,  
      "debitos": 0,  
      "creditos": 249375,  
      "saldo_actual": -1124375,  
      "base_retencion": 7125000,  
      "detalles": [  
        {  
          "fecha_documento": "2025-01-31",  
          "consecutivo_documento": "EG-3985",  
          "detalle_documento": "Retención arrendamiento enero Local 3",
```

```
        "debitos": 0,  
        "creditos": 157500,  
        "base_retencion": 4500000  
    },  
    {  
        "fecha_documento": "2025-02-28",  
        "consecutivo_documento": "EG-4050",  
        "detalle_documento": "Retención arrendamiento febrero Local 3",  
        "debitos": 0,  
        "creditos": 91875,  
        "base_retencion": 2625000  
    }  
]  
  
}  
  
}
```

Estructura principal de la respuesta

Campo	Tipo	Descripción
error	boolean	Estado de la respuesta. <code>false</code> para exitoso, <code>true</code> cuando ocurre un error.
message	string	Cadena vacía cuando la respuesta es exitosa. Si no hay resultados, contiene un mensaje informativo. Si hay error, describe el problema presentado.
body	array null	Listado de cuentas contables con sus movimientos. <code>null</code> cuando no hay resultados o hay un error.

Campos de cada cuenta (body)

Cada elemento dentro del arreglo `body` representa una cuenta contable y contiene las siguientes claves:

Campo	Tipo	Descripción
cuenta	string	Código numérico de la cuenta en el Plan Único de Cuentas (PUC).
nombre_cuenta	string	Nombre descriptivo de la cuenta PUC.

Campo	Tipo	Descripción
saldo_anterior	number	Saldo acumulado de la cuenta antes de la fecha inicial del rango consultado. Puede ser negativo.
debitos	number	Suma total de los débitos de la cuenta en el período consultado.
creditos	number	Suma total de los créditos de la cuenta en el período consultado.
saldo_actual	number	Saldo resultante de la cuenta al final del período. El cálculo varía según la clase de cuenta: para activos, gastos y costos (clases 1, 5 y 6) es $\text{saldo_anterior} + \text{débitos} - \text{créditos}$; para las demás clases es $\text{saldo_anterior} + \text{créditos} - \text{débitos}$. Puede ser negativo.
base_retencion	number	Valor acumulado de la base de retención de la cuenta en el período.
terceros	object	Objeto que contiene los terceros vinculados a la cuenta. Cada clave del objeto es el <code>tercero_id</code> del sistema (o <code>"sin_tercero"</code> si el movimiento no tiene tercero asociado).

Campos de cada tercero (terceros)

Cada entrada dentro del objeto `terceros` contiene las siguientes claves:

Campo	Tipo	Descripción
documento_tercero	string	Número de documento de identidad del tercero. Si no tiene tercero asociado, muestra <code>"Sin tercero"</code> .
nombre_tercero	string	Nombre completo del tercero. Si no tiene tercero asociado, muestra <code>"Sin tercero"</code> .
saldo_anterior	number	Saldo acumulado del tercero en esta cuenta antes de la fecha inicial. Puede ser negativo.
debitos	number	Suma total de los débitos del tercero en esta cuenta durante el período.
creditos	number	Suma total de los créditos del tercero en esta cuenta durante el período.
saldo_actual	number	Saldo resultante del tercero en esta cuenta al final del período. Mismo criterio de cálculo que el saldo de la cuenta.

Campo	Tipo	Descripción
base_retencion	number	Valor acumulado de la base de retención del tercero en esta cuenta.
detalles	array	Listado de documentos contables (movimientos individuales) del tercero en esta cuenta durante el período.

Campos de cada detalle (detalles)

Cada elemento dentro del arreglo `detalles` representa un documento contable individual:

Campo	Tipo	Descripción
fecha_documento	string	Fecha del documento contable en formato <code>YYYY-MM-DD</code> .
consecutivo_documento	string	Consecutivo del documento contable con su prefijo, en formato <code>PREFIJO-NÚMERO</code> (ej. <code>RC-10450</code> , <code>FV-8901</code> , <code>EG-4021</code>).
detalle_documento	string	Descripción u observaciones del documento contable.
debitos	number	Valor del débito del documento contable.
creditos	number	Valor del crédito del documento contable.
base_retencion	number	Valor de la base de retención del documento contable.

Nota sobre resultados vacíos

Si no existen movimientos contables para los parámetros indicados, la respuesta tendrá HTTP `200` con `error: false`, `body: null` y un `message` informativo: "No se encontraron registros con los parámetros indicados."

Nota sobre el cálculo de saldos

El cálculo del `saldo_actual` depende de la clase contable de la cuenta (primer dígito del código PUC):

— **Clases 1 (Activos), 5 (Gastos) y 6 (Costos):** `saldo_anterior + débitos - créditos`

— **Clases 2 (Pasivos), 3 (Patrimonio), 4 (Ingresos) y demás:**

`saldo_anterior + créditos - débitos`

Este comportamiento puede variar según la configuración del sistema

(parámetro interno de la plataforma).

4. Seguridad y Posibles Errores

¡Tu pase tiene fecha de caducidad!

Por medidas de seguridad, el `token` que te entregamos **solo dura 1 hora**. Una vez transcurrido ese tiempo, el pase expirará y el sistema te bloqueará el acceso devolviéndote un error `401`. Cuando esto ocurra, tu sistema simplemente debe volver a consumir el servicio de [Login](#) para pedir un pase nuevo y continuar trabajando.

Así se ve el error que te devolverá el sistema cuando tu token se haya vencido o intentes usar un pase inválido:

```
{
  "statusCode": 401,
  "error": {
    "type": "SERVER_ERROR",
    "description": "JWT Token expired."
  }
}
```

Si alguno de los parámetros no cumple con las validaciones, el sistema devolverá un error `400`. Por ejemplo:

```
{
  "error": true,
  "message": "La fecha inicial no tiene un formato válido (Y-m-d) o no es una fecha válida.",
  "body": null
}
```

Posibles errores:

Código HTTP	Descripción
-------------	-------------

400	Token faltante o inválido. Posibles causas: — No se envió el encabezado <code>Authorization</code>. Mensaje: <code>"JWT Token required."</code> — El encabezado no tiene el formato <code>Bearer {token}</code>. Mensaje: <code>"JWT Token not send."</code> — El token no fue encontrado en el sistema. Mensaje: <code>"JWT Token not found."</code> Parámetros inválidos. Posibles causas: — <code>fecha_ini</code> no tiene formato <code>YYYY-MM-DD</code> válido. Mensaje: <code>"La fecha inicial no tiene un formato válido (Y-m-d) o no es una fecha válida."</code> — <code>fecha_fin</code> no tiene formato <code>YYYY-MM-DD</code> válido. Mensaje: <code>"La fecha final no tiene un formato válido (Y-m-d) o no es una fecha válida."</code> — <code>fecha_ini</code> es posterior a <code>fecha_fin</code>. Mensaje: <code>"La fecha inicial debe ser menor o igual que la fecha final."</code> — <code>cuenta_ini</code> no es numérico. Mensaje: <code>"La cuenta inicial debe ser un valor numérico."</code> — <code>cuenta_fin</code> no es numérico. Mensaje: <code>"La cuenta final debe ser un valor numérico."</code> — <code>cuenta_ini</code> es mayor que <code>cuenta_fin</code>. Mensaje: <code>"La cuenta inicial debe ser menor o igual que la cuenta final."</code> — <code>cuenta_ini</code> es menor que 1. Mensaje: <code>"La cuenta inicial debe ser mayor o igual que 1."</code>
401	El token ha expirado. Debes generar uno nuevo consumiendo el servicio de Login. Mensaje: <code>"JWT Token expired."</code>
403	El cliente OAuth no tiene el scope necesario para esta operación. Para endpoints GET se requiere el scope <code>read</code>. Mensaje: <code>"Insufficient scope. Required: 'read', granted: '{scope_actual}'."</code>
500	Error interno del servidor al procesar la consulta del auxiliar general. Mensaje: <code>"Ocurrió un error al tratar de obtener el auxiliar general."</code>

5. Ejemplos de integración

Aquí tienes ejemplos de código listos para que tus desarrolladores los adapten a tu plataforma:

cURL

```
# Consultar auxiliar general con parámetros por defecto (año actual, todas las cuentas)
curl -X GET "https://{{instancia}}/service/v2/public/accounting/general-ledger" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer TU_TOKEN_AQUI"
```

```
# Consultar auxiliar general de enero a junio 2025, cuentas de activos y pasivos (1 a 2)
curl -X GET "https://{{instancia}}/service/v2/public/accounting/general-
ledger?fecha_ini=2025-01-01&fecha_fin=2025-06-30&cuenta_ini=1&cuenta_fin=2" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer TU_TOKEN_AQUI"
```

PHP

```
<?php

$instance = 'tu_instancia'; // Reemplaza con tu instancia real
$token = 'TU_TOKEN_AQUI'; // Token obtenido del servicio Login

// Parámetros de consulta
$fecha_ini = '2025-01-01';
$fecha_fin = '2025-06-30';
$cuenta_ini = '1';
$cuenta_fin = '4';

$queryParams = http_build_query([
    'fecha_ini' => $fecha_ini,
    'fecha_fin' => $fecha_fin,
    'cuenta_ini' => $cuenta_ini,
    'cuenta_fin' => $cuenta_fin
]);

$url = "https://{ $instance }/service/v2/public/accounting/general-ledger?{$queryParams}";

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    'Content-Type: application/json',
    "Authorization: Bearer {$token}"
]);

$response = curl_exec($ch);
```

```

$http_code = curl_getinfo($ch, CURLINFO_HTTP_CODE);

if (curl_errno($ch)) {
    echo 'Error: ' . curl_error($ch);
} else {
    echo "Código de estado HTTP: " . $http_code . "\n";

    $info = json_decode($response, true);
    $body = $info['body'] ?? [];

    if ($http_code === 200 && !empty($body)) {
        echo "Se consultaron " . count($body) . " cuentas contables.\n\n";

        foreach ($body as $cuenta) {
            echo "Cuenta: {$cuenta['cuenta']} - {$cuenta['nombre_cuenta']}\n";
            echo "    Saldo anterior: $" . number_format($cuenta['saldo_anterior'], 2) .
"\n";

            echo "    Débitos: $" . number_format($cuenta['debitos'], 2) . "\n";
            echo "    Créditos: $" . number_format($cuenta['creditos'], 2) . "\n";
            echo "    Saldo actual: $" . number_format($cuenta['saldo_actual'], 2) . "\n";

            foreach ($cuenta['terceros'] as $terceroId => $tercero) {
                echo "        Tercero: {$tercero['documento_tercero']} -
{$tercero['nombre_tercero']}\n";
                echo "        Movimientos: " . count($tercero['detalles']) . "\n";
            }
            echo "\n";
        }
    } else {
        echo "Mensaje: " . ($info['message'] ?? 'Sin resultados') . "\n";
    }
}
curl_close($ch);

?>

```

Python

```
import requests

# 1. Configura tus datos de acceso
instancia = 'mi-inmobiliaria.nuby.app' # Reemplaza con tu dirección web completa
token = 'TU_TOKEN_AQUI' # Token obtenido del servicio Login

# 2. Prepara la dirección y los parámetros
url = f"https://{instancia}/service/v2/public/accounting/general-ledger"

params = {
    "fecha_ini": "2025-01-01",
    "fecha_fin": "2025-06-30",
    "cuenta_ini": "1",
    "cuenta_fin": "4"
}

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {token}"
}

# 3. Envía la petición GET y procesa la respuesta
try:
    response = requests.get(url, params=params, headers=headers)

    print(f"Código de estado HTTP: {response.status_code}")

    if response.status_code == 200:
        info = response.json()
        body = info.get('body') or []

        if body:
            print(f"Cuentas encontradas: {len(body)}\n")
            for cuenta in body:
                print(f"Cuenta: {cuenta['cuenta']} - {cuenta['nombre_cuenta']}")
                print(f" Saldo anterior: ${cuenta['saldo_anterior']:, .2f}")
                print(f" Débitos: ${cuenta['debitos']:, .2f}")
                print(f" Créditos: ${cuenta['creditos']:, .2f}")
                print(f" Saldo actual: ${cuenta['saldo_actual']:, .2f}")
```

```

        for tercero_id, tercero in cuenta['terceros'].items():
            print(f"    Tercero [{tercero_id}]: {tercero['documento_tercero']} - {tercero['nombre_tercero']}")
            print(f"        Movimientos: {len(tercero['detalles'])}")
            print()
        else:
            print(f"Mensaje: {info.get('message', 'Sin resultados')}")
    else:
        print("Error al consultar el auxiliar general.")
        print(f"Detalle del error: {response.text}")

except Exception as e:
    print(f"Ocurrió un error de conexión: {e}")

```

JavaScript

```

// 1. Configura tus datos de acceso
const instancia = 'mi-inmobiliaria.nuby.app'; // Reemplaza con tu dirección web completa
const token = 'TU_TOKEN_AQUI'; // Token obtenido del servicio Login

// 2. Prepara la dirección con parámetros
const params = new URLSearchParams({
    fecha_ini: '2025-01-01',
    fecha_fin: '2025-06-30',
    cuenta_ini: '1',
    cuenta_fin: '4'
});

const url = `https://${instancia}/service/v2/public/accounting/general-ledger?${params}`;

// 3. Función para consultar el auxiliar general
async function consultarAuxiliarGeneral() {
    try {
        const response = await fetch(url, {
            method: 'GET',
            headers: {

```

```

        'Content-Type': 'application/json',
        'Authorization': `Bearer ${token}`
    }
});

console.log(`Código de estado HTTP: ${response.status}`);

if (response.ok) {
    const info = await response.json();
    const body = info.body || [];

    if (body.length > 0) {
        console.log(`Cuentas encontradas: ${body.length}\n`);
        body.forEach(cuenta => {
            console.log(`Cuenta: ${cuenta.cuenta} - ${cuenta.nombre_cuenta}`);
            console.log(` Saldo anterior:
${cuenta.saldo_anterior.toLocaleString()}`);
            console.log(` Débitos: ${cuenta.debitos.toLocaleString()}`);
            console.log(` Créditos: ${cuenta.creditos.toLocaleString()}`);
            console.log(` Saldo actual:
${cuenta.saldo_actual.toLocaleString()}`);

            for (const [terceroId, tercero] of Object.entries(cuenta.terceros)) {
                console.log(` Tercero [${terceroId}]:
${tercero.documento_tercero} - ${tercero.nombre_tercero}`);
                console.log(` Movimientos: ${tercero.detalles.length}`);
            }
            console.log('');
        });
    } else {
        console.log(`Mensaje: ${info.message || 'Sin resultados'}`);
    }
} else {
    console.log('Error al consultar el auxiliar general.');
    const errorData = await response.text();
    console.log(`Detalle del error: ${errorData}`);
}
} catch (error) {
    console.error(`Ocurrió un error de conexión: ${error}`);
}

```

```
}  
  
}  
  
// 4. Ejecutamos la función  
consultarAuxiliarGeneral();
```

Power Query M (Excel / Power BI)

```
let  
    // 1. Configura tus datos de acceso  
    instancia = "mi-inmobiliaria.nuby.app", // Reemplaza con tu dirección web completa  
    token = "TU_TOKEN_AQUI", // Token obtenido del servicio Login  
  
    // 2. Prepara la dirección con parámetros  
    url = "https://" & instancia & "/service/v2/public/accounting/general-  
ledger?fecha_ini=2025-01-01&fecha_fin=2025-06-30&cuenta_ini=1&cuenta_fin=4",  
  
    // 3. Envía la petición GET  
    response = Web.Contents(url, [  
        Headers = [  
            #"Content-Type" = "application/json",  
            #"Authorization" = "Bearer " & token  
        ]  
    ]),  
  
    // 4. Decodifica la respuesta JSON  
    jsonResponse = Json.Document(response),  
    body = jsonResponse[body],  
  
    // 5. Convierte el listado de cuentas en tabla  
    tablaCuentas = Table.FromList(body, Splitter.SplitByNothing(), null, null,  
ExtraValues.Error),  
    expandidoCuentas = Table.ExpandRecordColumn(tablaCuentas, "Column1",  
        {"cuenta", "nombre_cuenta", "saldo_anterior", "debitos", "creditos",  
"saldo_actual", "base_retencion", "terceros"},  
        {"Cuenta", "NombreCuenta", "SaldoAnterior", "Debitos", "Creditos", "SaldoActual",  
"BaseRetencion", "Terceros"}),
```

```
// 6. (Opcional) Expandir terceros para obtener detalle por tercero
// Nota: La estructura de terceros es un objeto con claves dinámicas.
// Para un análisis más detallado, es recomendable procesar los terceros
// con Record.FieldValues y luego expandir los detalles.
tercerosList = Table.AddColumn(expandidoCuentas, "TercerosLista", each
Record.FieldValues([Terceros])),
    expandidoTerceros = Table.ExpandListColumn(tercerosList, "TercerosLista"),
    expandidoTercerosCampos = Table.ExpandRecordColumn(expandidoTerceros, "TercerosLista",
        {"documento_tercero", "nombre_tercero", "saldo_anterior", "debitos", "creditos",
"saldo_actual", "base_retencion"},
        {"DocTercero", "NombreTercero", "SaldoAntTercero", "DebitosTercero",
"CreditosTercero", "SaldoActTercero", "BaseRetTercero"})
in
    expandidoTercerosCampos
```