Esta evidencia se centra en la apropiación del conocimiento requerido para la creación y manipulación de

bases de datos no relacionales, utilizando SQL exclusivamente.

Deberá utilizar un script, ejecutarlo en un motor de bases de datos MySQL o MSSQL según su

preferencia y resolver algunas consultas para extraer o manipular información, usando sentencias DDL o

DML según sea el caso. Los detalles del modelo de datos físicos y las consultas a desarrollar las

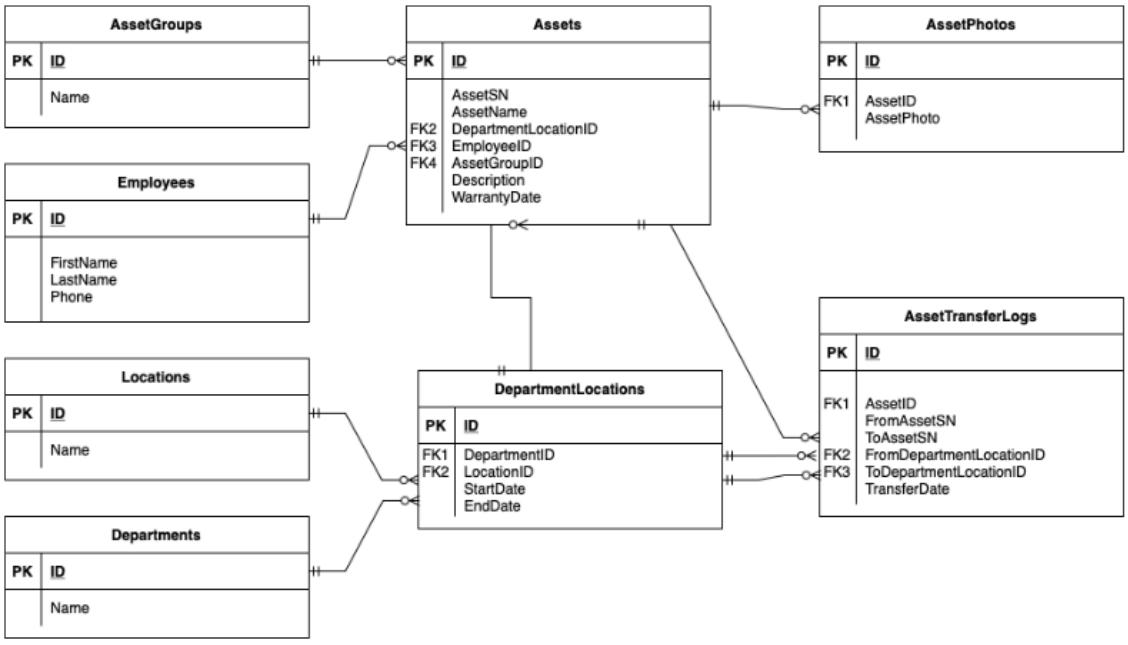
encontrará en el siguiente problema:

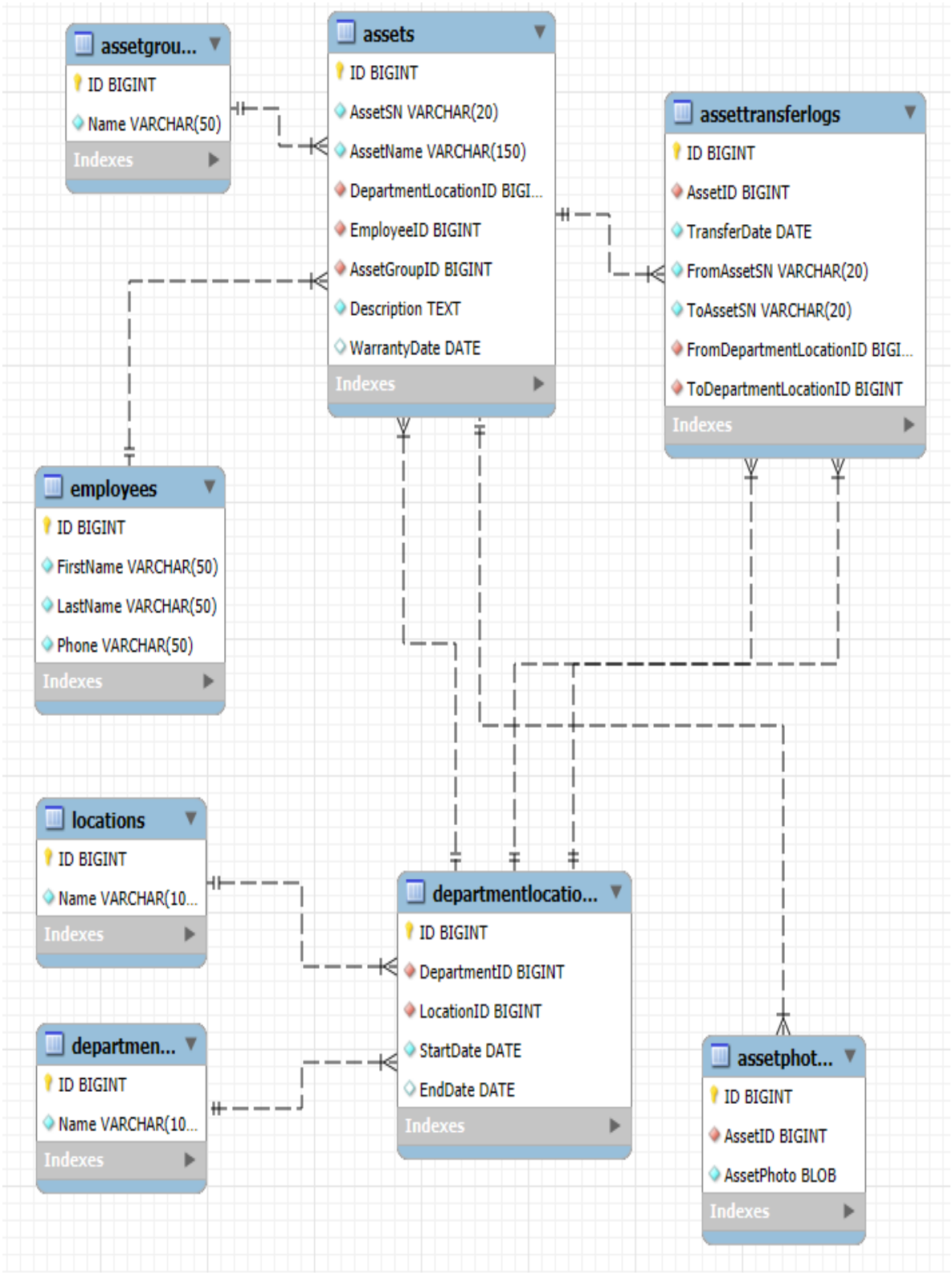
Una empresa en constante crecimiento del sector petrolero ha venido experimentando buenos resultados

financieros, que le permiten iniciar proyectos de expansión en diferentes localidades e incluso fuera de su

país de origen, por lo cual requiere organizar toda la información existente referente a su inventario y

activos. Como resultado de esta necesidad se desarrolló el siguiente modelo de datos:

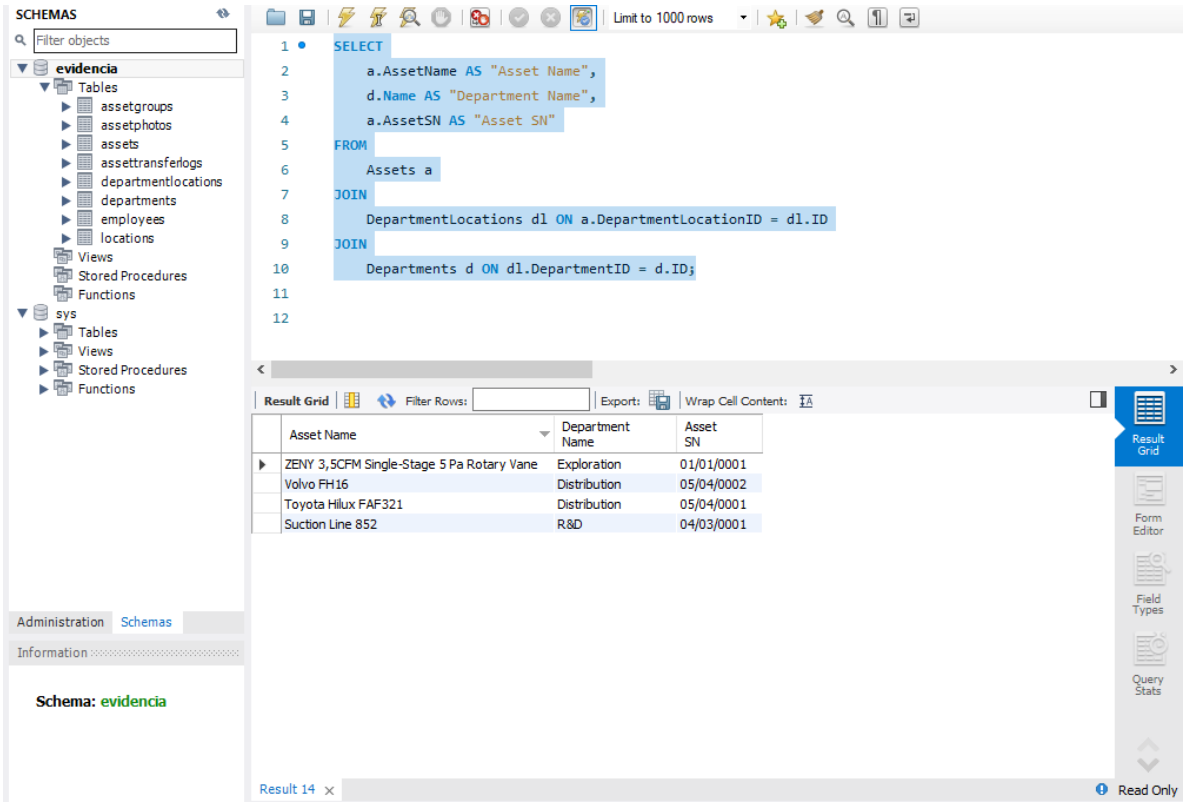



assetgrou...
ID BIGINT
Name VARCHAR(50)
Indexes
assets
ID BIGINT
AssetSN VARCHAR(20)
AssetName VARCHAR(150)
DepartmentLocationID BIGI...
EmployeeID BIGINT
AssetGroupID BIGINT
Description TEXT
WarrantyDate DATE
Indexes
assettransferlogs
ID BIGINT
AssetID BIGINT
TransferDate DATE
FromAssetSN VARCHAR(20)
ToAssetSN VARCHAR(20)
FromDepartmentLocationID BIGI...
ToDepartmentLocationID BIGINT
Indexes
employees
ID BIGINT
FirstName VARCHAR(50)
LastName VARCHAR(50)
Phone VARCHAR(50)
Indexes
locations
ID BIGINT
Name VARCHAR(10...
Indexes
departmentlocatio...
ID BIGINT
DepartmentID BIGINT
LocationID BIGINT
StartDate DATE
EndDate DATE
Indexes
departmen...
ID BIGINT
Name VARCHAR(10...
Indexes
assetphot...
ID BIGINT
AssetID BIGINT
AssetPhoto BLOB
Indexes

1. Crear una consulta que permita obtener el catálogo de activos. Cada registro deberá tener los campos Asset name, Department name, Asset SN.

CODIGO

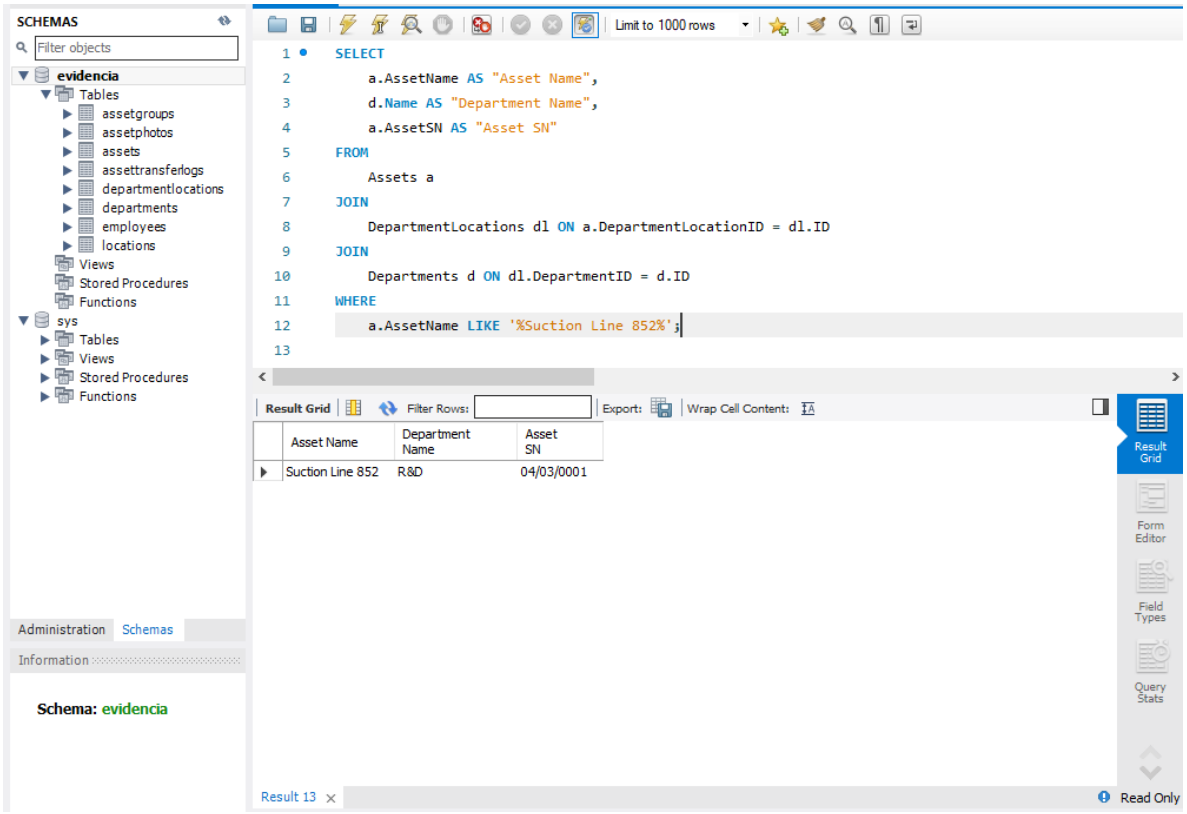
```sql
SELECT
    a.AssetName AS "Asset Name",
    d.Name AS "Department Name",
    a.AssetSN AS "Asset SN"
FROM
    Assets a
JOIN
    DepartmentLocations dl ON a.DepartmentLocationID = dl.ID
JOIN
    Departments d ON dl.DepartmentID = d.ID;
```



2. Crear una consulta que permita filtrar el catálogo de activos del punto anterior a partir de la búsqueda aproximada de una palabra clave del Asset name

CODIGO

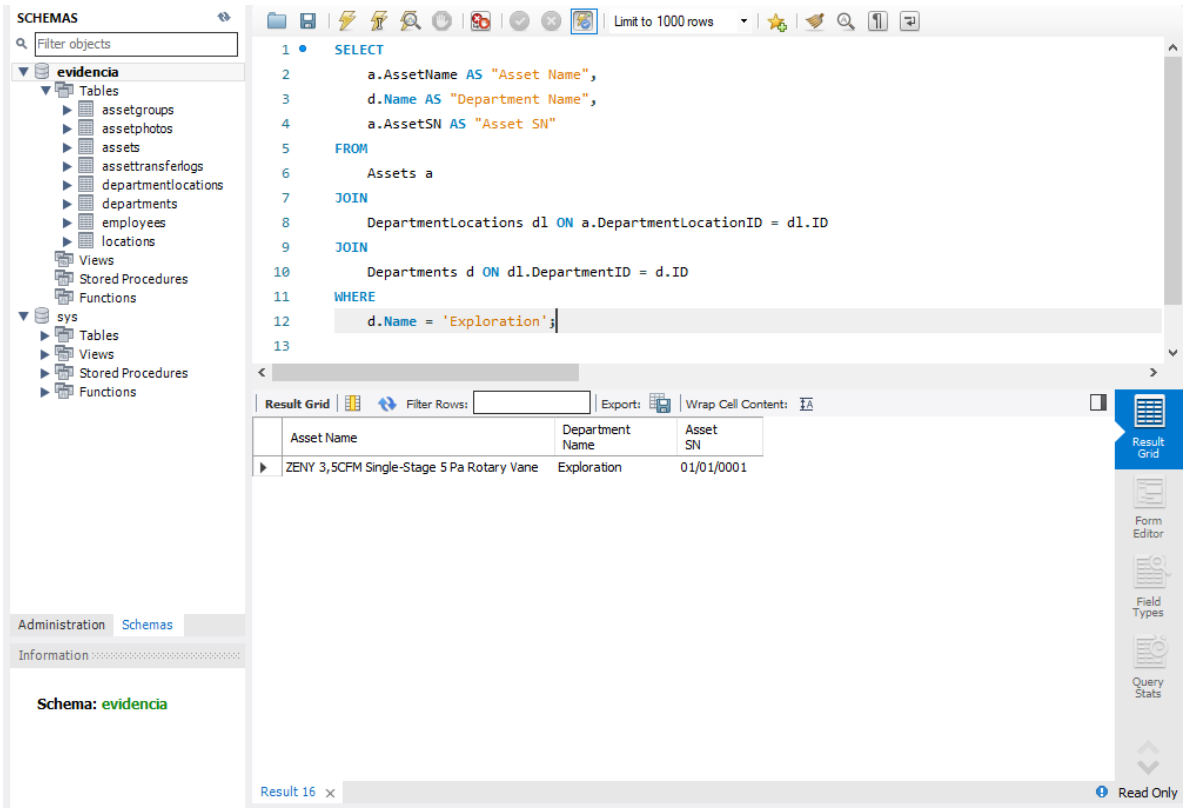
```
SELECT
    a.AssetName AS "Asset Name",
    d.Name AS "Department Name",
    a.AssetSN AS "Asset SN"
FROM
    Assets a
JOIN
    DepartmentLocations dl ON a.DepartmentLocationID = dl.ID
JOIN
    Departments d ON dl.DepartmentID = d.ID
WHERE
    a.AssetName LIKE '%Suction Line 852%';
```



3. Crear una consulta que permita filtrar el catálogo de activos del punto anterior a partir de la búsqueda exacta del Department name.

CODIGO

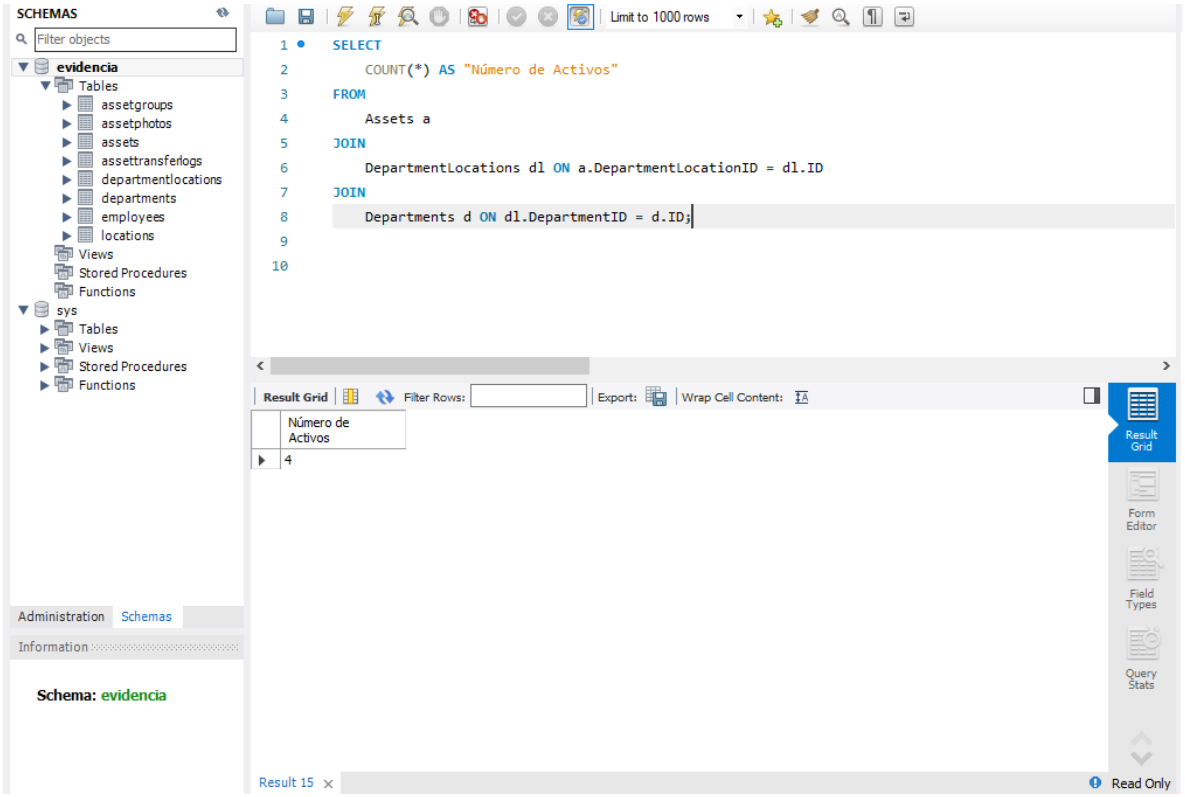
```sql
SELECT
    a.AssetName AS "Asset Name",
    d.Name AS "Department Name",
    a.AssetSN AS "Asset SN"
FROM
    Assets a
JOIN
    DepartmentLocations dl ON a.DepartmentLocationID = dl.ID
JOIN
    Departments d ON dl.DepartmentID = d.ID
WHERE
    d.Name = 'Exploration';
```



4. Crear una consulta que devuelva como resultado el número de registros encontrados en el catálogo de activos

CODIGO

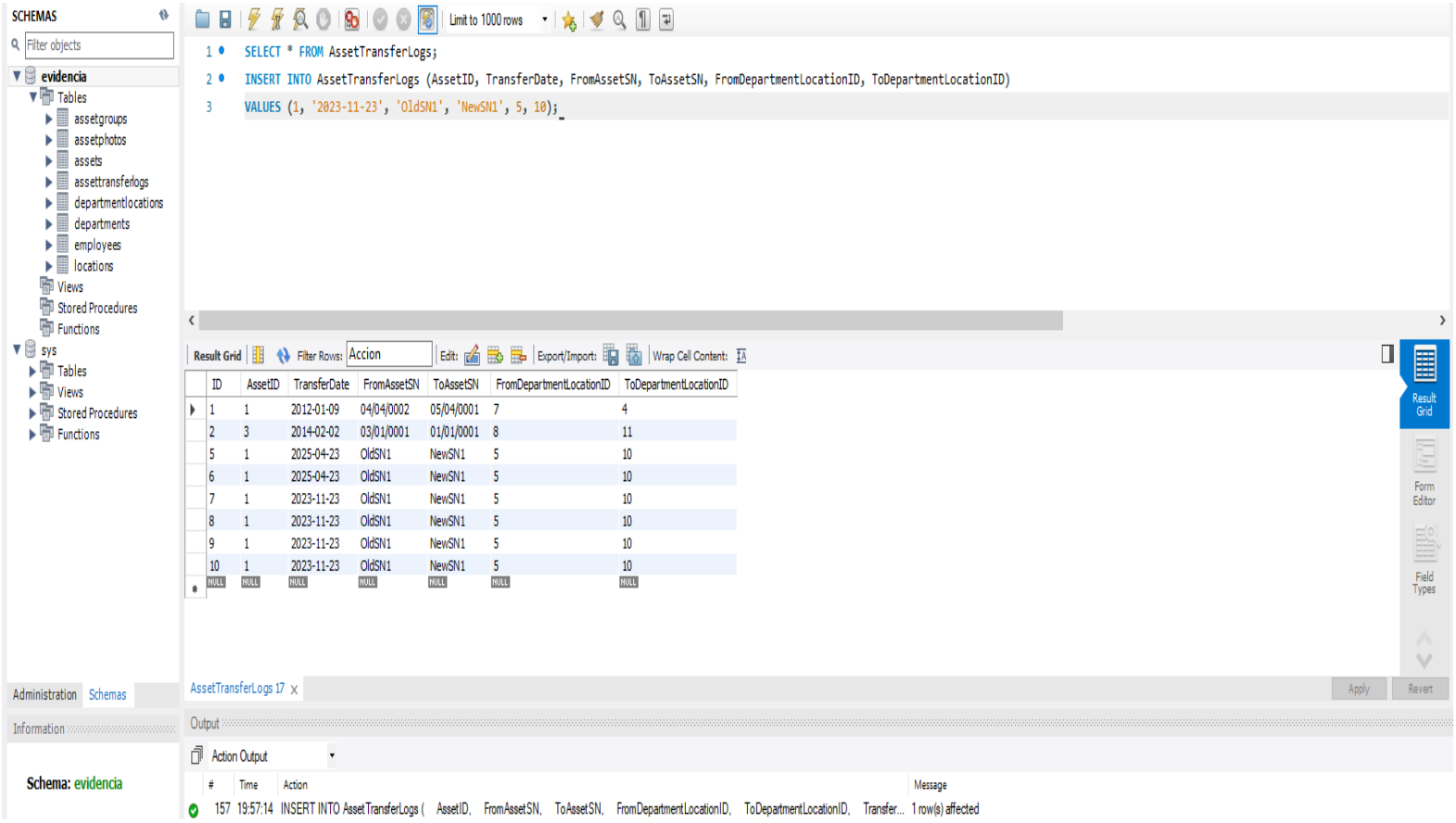
```
SELECT
    COUNT(*) AS "Número de Activos"
FROM
    Assets a
JOIN
    DepartmentLocations dl ON a.DepartmentLocationID = dl.ID
JOIN
    Departments d ON dl.DepartmentID = d.ID;
```



5. El modelo está diseñado para permitir la transferencia de activos entre departamentos, crear un conjunto de sentencias que ejemplifiquen cómo se registraría esta transacción en la base de datos diseñada (debe usar la tabla AssetTransferLogs)

CODIGO

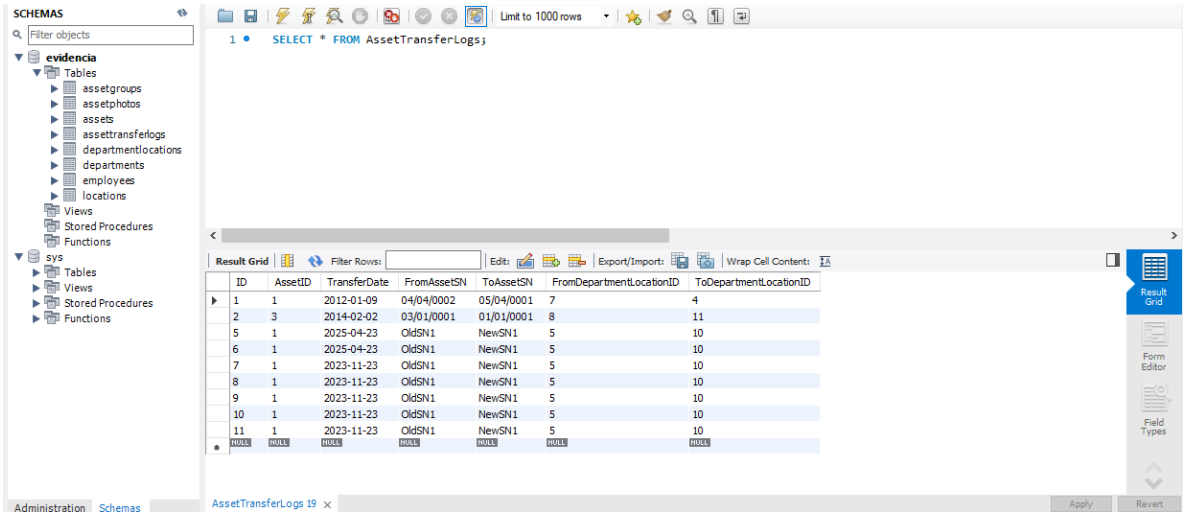
```
SELECT * FROM AssetTransferLogs;
INSERT INTO AssetTransferLogs (AssetID, TransferDate, FromAssetSN,
ToAssetSN, FromDepartmentLocationID, ToDepartmentLocationID)
VALUES (1, '2023-11-23', 'OldSN1', 'NewSN1', 5, 10);
```



6. Crear una consulta que permita ver todo el catálogo de transferencias realizadas

CODIGO

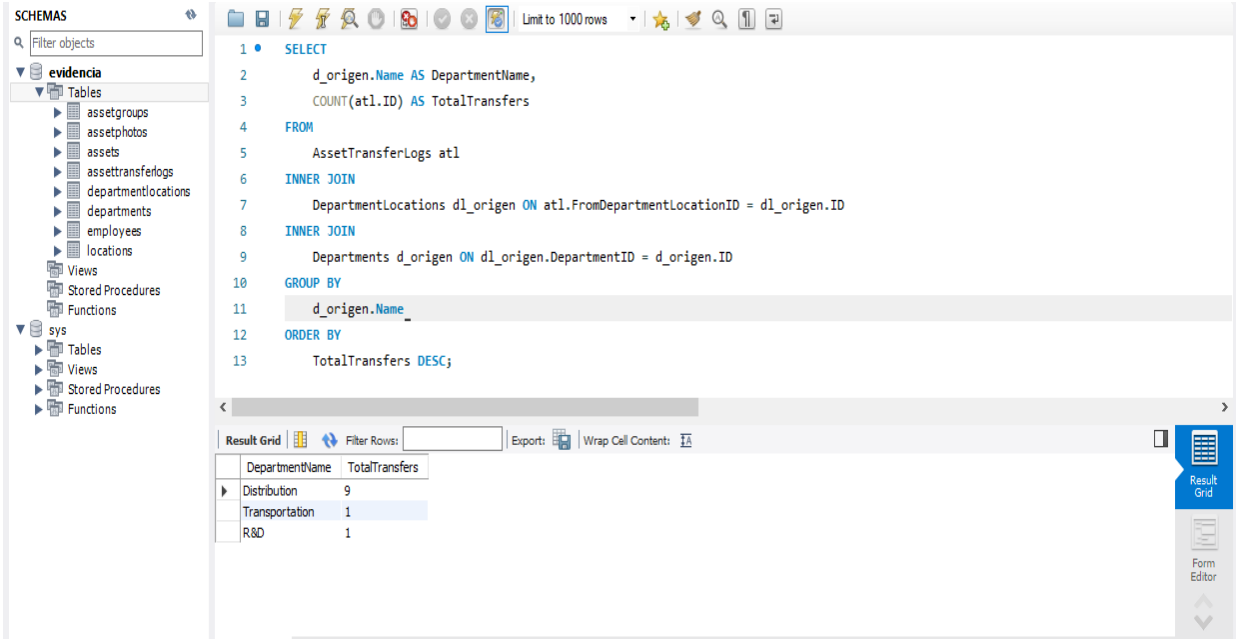
```
SELECT * FROM AssetTransferLogs;
```



7. Crear una consulta para obtener el nombre del departamento desde el cual se han realizado más transferencias.

CODIGO

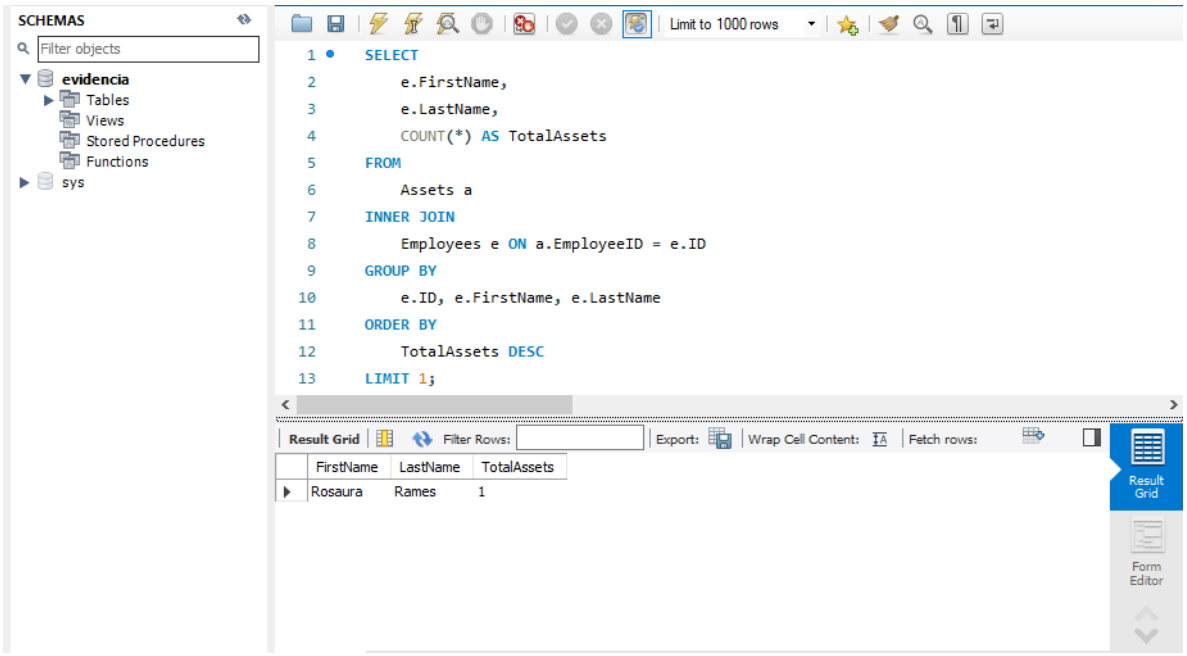
```
SELECT
    d_origen.Name AS DepartmentName,
    COUNT(atl.ID) AS TotalTransfers
FROM
    AssetTransferLogs atl
INNER JOIN
    DepartmentLocations dl_origen ON atl.FromDepartmentLocationID =
dl_origen.ID
INNER JOIN
    Departments d_origen ON dl_origen.DepartmentID = d_origen.ID
GROUP BY
    d_origen.Name
ORDER BY
    TotalTransfers DESC;
```



8. Obtener los datos del empleado que más activos tiene asignados.

CODIGO

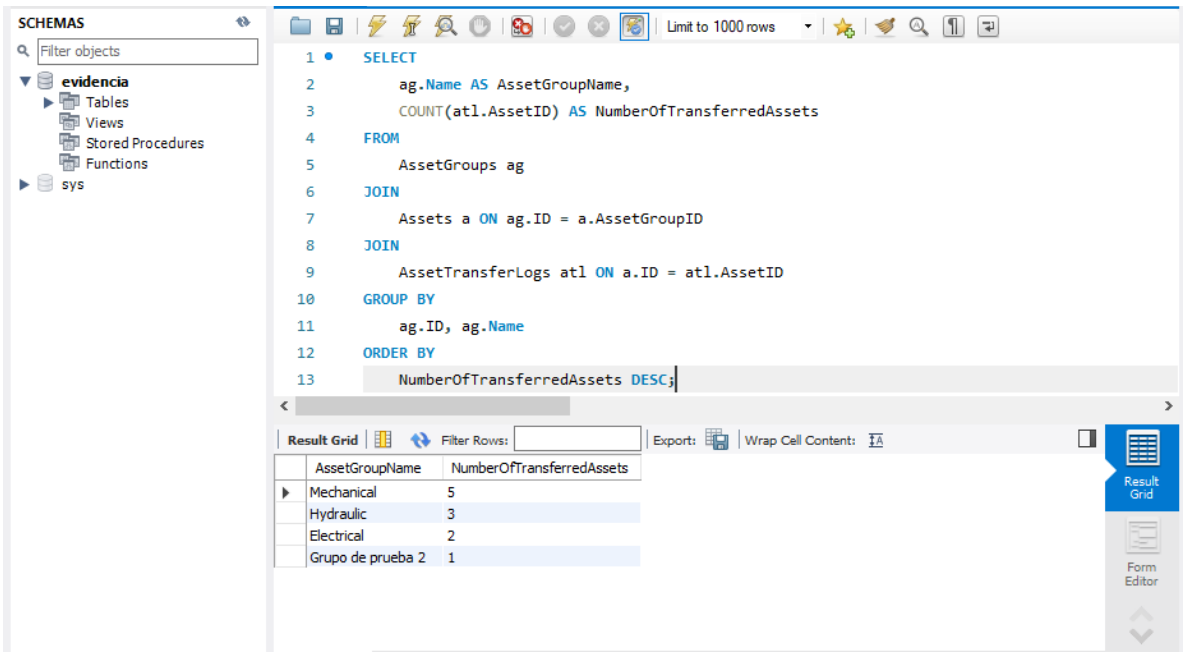
```sql
SELECT
    e.FirstName,
    e.LastName,
    COUNT(*) AS TotalAssets
FROM
    Assets a
INNER JOIN
    Employees e ON a.EmployeeID = e.ID
GROUP BY
    e.ID, e.FirstName, e.LastName
ORDER BY
    TotalAssets DESC
LIMIT 1;
```



9. Obtener los grupos de activos y las cantidades de activos que están relacionados en el AssetTransferLogs.

CODIGO

```sql
SELECT
    ag.Name AS AssetGroupName,
    COUNT(atl.AssetID) AS NumberOfTransferredAssets
FROM
    AssetGroups ag
JOIN
    Assets a ON ag.ID = a.AssetGroupID
JOIN
    AssetTransferLogs atl ON a.ID = atl.AssetID
GROUP BY
    ag.ID, ag.Name
ORDER BY
    NumberOfTransferredAssets DESC;
```



10. Cantidad de activos que no han estado relacionados en transferencias.

CODIGO

```
SELECT

    COUNT(a.ID) AS NumberOfAssetsWithoutTransfers
FROM
    Assets a
LEFT JOIN
    AssetTransferLogs atl ON a.ID = atl.AssetID
WHERE
    atl.AssetID IS NULL;
```