Vistas

Una vista en SQL Server es un objeto de base de datos que representa una consulta de selección almacenada.

Es como una tabla virtual: no guarda datos propios, sino que muestra datos provenientes de una o varias tablas reales.

Sirve para:

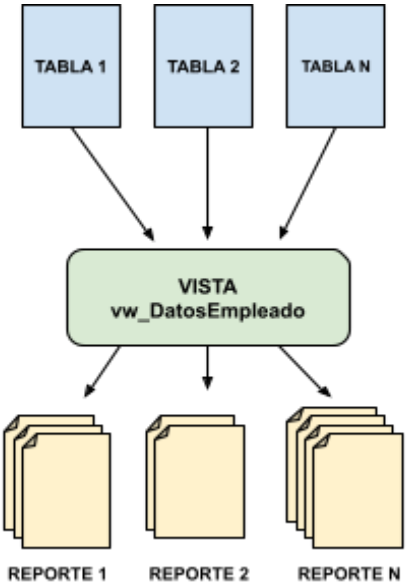
- Simplificar consultas complejas.
- Dar acceso controlado a ciertos datos.
- Mejorar la seguridad ocultando columnas sensibles.
- Estandarizar consultas comunes.

Sintaxis básica

```
CREATE VIEW nombreVista AS
SELECT columnas
FROM tablas
WHERE condiciones;
```

Las vistas pueden tener Joins, Funciones de resumen y condiciones para filtrar datos. Luego se podrán realizar consultas de selección a la vista creada para obtener información.

Supongamos que debemos obtener los datos de los empleados, junto con su área, la antigüedad en el mismo y el último sueldo cobrado. Tengamos en cuenta que el cálculo de dicha información puede resultar un poco compleja y rebuscada. Si necesitáramos en muchos reportes y listados obtener éstos registros, sería más sencillo crear una vista que se encargue de obtener éstos datos y luego cada vez que se necesite acceder a ellos realizar una consulta de selección a la vista.

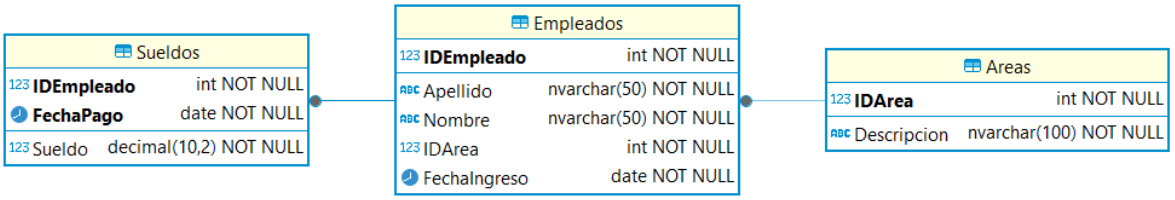


Como se puede observar en la imagen, los reportes 1, 2 y N surgen a partir de la vista vw_DatosEmpleado. Por otra parte, la vista proviene de las tablas 1, 2 y N. Como habíamos explicado, el cálculo de las columnas antigüedad y sueldo podrían llegar a ser complejos de calcular.

Mediante el uso de la vista lo que conseguimos es lograr abstraer la complejidad. Ya que al momento de elaborar los reportes se solicitarán los datos directamente de nuestra vista sin preocuparnos por el cálculo de las columnas.

Por otra parte, si en algún momento la empresa decidiera modificar el proceso de cálculo de sueldo, sólo bastará con modificar el código SQL que compone a la vista, replicando automáticamente el cambio a todos los reportes, simplificando así la tarea.

Supongamos que la base de datos es la siguiente:



La vista para simplificar el listado mencionado sería la siguiente:

```
CREATE VIEW VW_DatosEmpleado AS
SELECT
	e.IDEmpleado,
	e.Apellido,
	e.Nombre,
	a.Descripcion AS NombreArea,
	DATEDIFF(YEAR, e.FechaIngreso, GETDATE())
	- CASE
	WHEN DATEADD(YEAR, DATEDIFF(YEAR, e.FechaIngreso, GETDATE()),
e.FechaIngreso) > GETDATE()
	THEN 1
	ELSE 0
	END AS AntiguedadAnios,
	(
	SELECT TOP 1 s.Sueldo
	FROM Sueldos s
	WHERE s.IDEmpleado = e.IDEmpleado
	ORDER BY s.FechaPago DESC
	) AS UltimoSueldo
FROM
	Empleados e
INNER JOIN
	Areas a ON e.IDArea = a.IDArea;
```

La consulta SQL crea un objeto en la base de datos llamado VW_DatosEmpleado. Al ser una vista se podrá realizar consultas de selección como si fuese una tabla.

Por lo que si ejecutamos:

```
SELECT * FROM VW_DatosEmpleado;
```

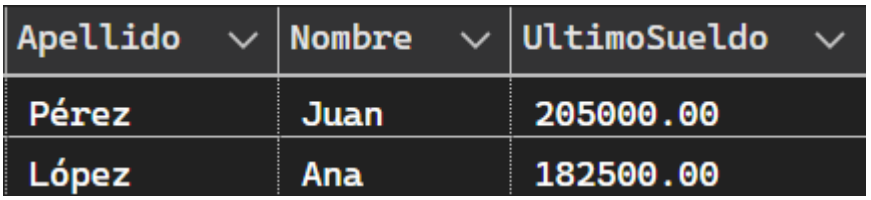
Obtendremos el siguiente resultado (la información dependerá de la fecha del sistema):

IDEmpleado	Apellido	Nombre	NombreArea	AntiguedadAnios	UltimoSueldo
101	González	María	Recursos Humanos	3	152000.00
102	Pérez	Juan	Sistemas	3	205000.00
103	López	Ana	Sistemas	2	182500.00
104	Fernández	Carlos	Contabilidad	4	172000.00
105	Ramírez	Lucía	Recursos Humanos	2	157000.00

La ventaja es que podremos utilizar la columna UltimoSueldo utilizando un simple WHERE como si fuese una columna real. Por ejemplo si ejecutamos:

```
SELECT Apellido, Nombre, UltimoSueldo
From VW_DatosEmpleado WHERE UltimoSueldo > 180000;
```

El resultado sería:



Apellido	Nombre	UltimoSueldo
Pérez	Juan	205000.00
López	Ana	182500.00

Podemos observar como las vistas simplifican el acceso a los datos encapsulando una consulta compleja de una manera simplificada.

Si queremos modificar una vista existente utilizaremos la sintaxis ALTER VIEW

```
ALTER VIEW VW_DatosEmpleado AS
SELECT
      e.IDEmpleado,
      e.Apellido,
      e.Nombre,
      a.Descripcion AS NombreArea
FROM
      Empleados e
INNER JOIN
      Areas a ON e.IDArea = a.IDArea;
```

Luego de ejecutar la consulta de ALTER VIEW, la vista obtendrá solamente el IDEmpleado, Apellido, Nombre y Descripción del nombre del área. Es decir, cambiará el resultado de lo que obtendrá cuando seleccionemos información de ella.

Por último si deseamos eliminar definitivamente una vista utilizaremos DROP VIEW

```
DROP VIEW VW_DatosEmpleado;
```

Posteriormente a ejecutar la consulta de DROP VIEW, la vista no existirá más en la base de datos.

Anexo de código SQL para creación de Base de Datos de Ejemplo

```sql
CREATE DATABASE EjemploVista
COLLATE Latin1_General_CI_AI
Go

Use EjemploVista
Go
CREATE TABLE Areas (
	IDArea INT PRIMARY KEY,
	Descripcion NVARCHAR(100) NOT NULL
);

CREATE TABLE Empleados (
	IDEmpleado INT PRIMARY KEY,
	Apellido NVARCHAR(50) NOT NULL,
	Nombre NVARCHAR(50) NOT NULL,
	IDArea INT NOT NULL,
	FechaIngreso DATE NOT NULL,
	FOREIGN KEY (IDArea) REFERENCES Areas(IDArea)
);

CREATE TABLE Sueldos (
	IDEmpleado INT NOT NULL,
	FechaPago DATE NOT NULL,
	Sueldo DECIMAL(10,2) NOT NULL,
	PRIMARY KEY (IDEmpleado, FechaPago),
	FOREIGN KEY (IDEmpleado) REFERENCES Empleados(IDEmpleado)
);

INSERT INTO Areas (IDArea, Descripcion) VALUES (1, 'Recursos Humanos');
INSERT INTO Areas (IDArea, Descripcion) VALUES (2, 'Sistemas');
INSERT INTO Areas (IDArea, Descripcion) VALUES (3, 'Contabilidad');

INSERT INTO Empleados (IDEmpleado, Apellido, Nombre, IDArea, FechaIngreso) VALUES
(101, 'González', 'María', 1, '2022-03-15');
INSERT INTO Empleados (IDEmpleado, Apellido, Nombre, IDArea, FechaIngreso) VALUES
(102, 'Pérez', 'Juan', 2, '2021-07-01');
INSERT INTO Empleados (IDEmpleado, Apellido, Nombre, IDArea, FechaIngreso) VALUES
(103, 'López', 'Ana', 2, '2023-01-10');
INSERT INTO Empleados (IDEmpleado, Apellido, Nombre, IDArea, FechaIngreso) VALUES
(104, 'Fernández', 'Carlos', 3, '2020-11-20');
INSERT INTO Empleados (IDEmpleado, Apellido, Nombre, IDArea, FechaIngreso) VALUES
(105, 'Ramírez', 'Lucía', 1, '2022-05-05');

INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (101, '2024-03-31',
150000.00);
INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (101, '2024-04-30',
152000.00);

INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (102, '2024-03-31',
200000.00);
INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (102, '2024-04-30',
```

```
205000.00);

INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (103, '2024-03-31',
180000.00);
INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (103, '2024-04-30',
182500.00);

INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (104, '2024-03-31',
170000.00);
INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (104, '2024-04-30',
172000.00);

INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (105, '2024-03-31',
155000.00);
INSERT INTO Sueldos (IDEmpleado, FechaPago, Sueldo) VALUES (105, '2024-04-30',
157000.00);
```