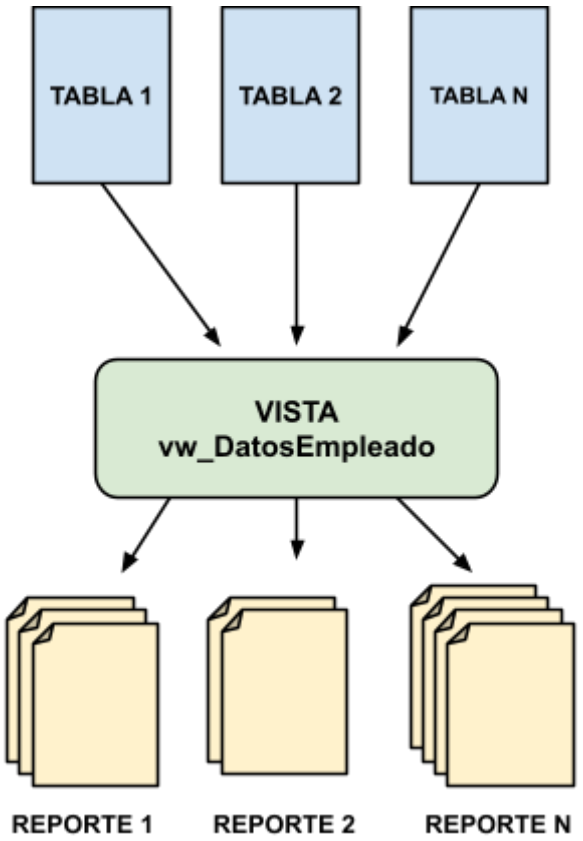
Carrera: Técnico Superior en Programación
Materia: Laboratorio de Computación III
Tema: Vistas

Vistas

Una vista consiste en una tabla virtual definida mediante una consulta del tipo SELECT. De ésta manera se obtiene dinámicamente el conjunto de columnas y filas de una tabla o de dos o más tablas relacionadas.
Las vistas ayudan a abstraer la complejidad de una consulta, simplificando la visión de la base de datos que tiene un usuario.

Las vistas nos ofrecen la posibilidad de:

Reutilizar instrucciones SELECT complejas
Supongamos que debemos obtener los datos de los empleados, junto con su cargo, la antigüedad en el mismo, el domicilio y el sueldo actual a cobrar. 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.

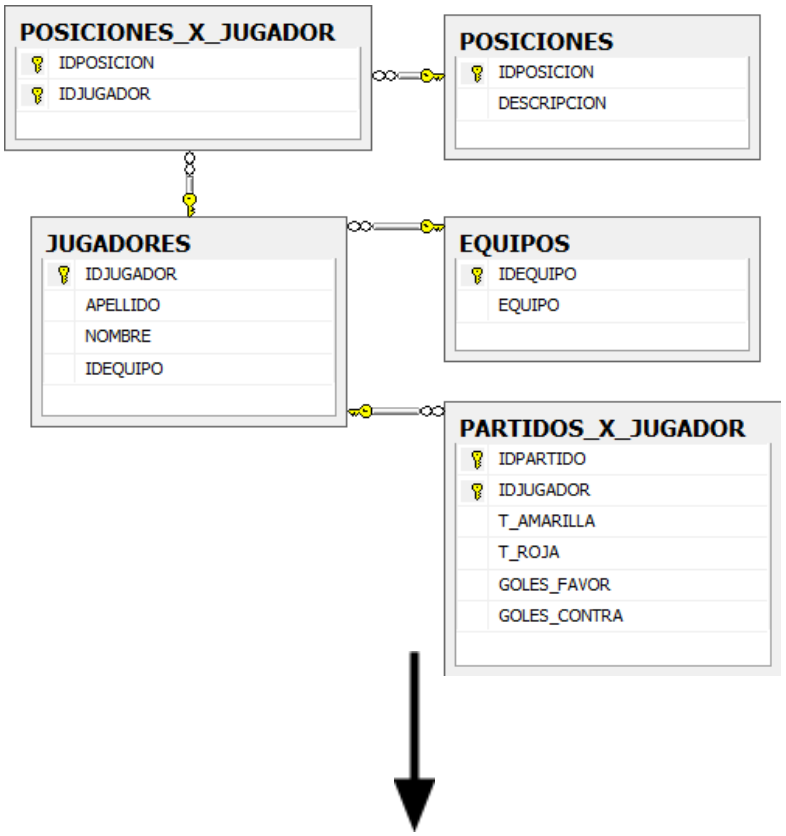


Como se puede observar en el gráfico de la izquierda, los reportes 1, 2 y N surgen a partir de la vista *vw_DatosEmpleado*. Por otra parte, la vista proviene de las tablas 1, 2 y N. Como habíamos explicado, el cálculo de las columnas antigüedad y sueldo podrían llegar a ser complejos de calcular.
Mediante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

Denormalizar datos

Si bien la estructura de nuestra base de datos permanecerá normalizada, al momento de utilizar vistas no será necesario que éstas se encuentren normalizadas. Por ejemplo, podría obtener el apellido, nombre, edad, domicilio y ciudad de residencia, nombre de la obra social y cantidad de ventas realizadas de un empleado. Este conjunto de datos proviene de diversas tablas que se encontrarán relacionadas mediante claves. Algunas de las columnas de la vista provienen de éstas tablas y otras provienen de conexiones entre tablas (JOINS), cálculos y resúmenes de datos.

Quien utilice la vista no deberá preocuparse por recorrer las conexiones o realizar los cálculos.



	APELLIDO	NOMBRE	EQUIPO	CANT_PARTIDOS	TOTAL_TARJETAS	TOTAL_GOLES	PROMEDIO_GOLES
1	GABBARINI	ADRIAN	INDEPENDIENTE	1	1	0	0.000000
2	MILITO	GABRIEL	INDEPENDIENTE	1	3	0	0.000000
3	PARRA	FACUNDO	INDEPENDIENTE	1	1	2	2.000000
4	VEGA	DIEGO	RIVER PLATE	1	1	0	0.000000
5	DOMINGUIEZ	CHORI	RIVER PLATE	1	2	1	1.000000
6	TREZEGUET	DAVID	RIVER PLATE	2	3	3	1.500000

Aplicar filtros
Como las vistas permiten que se les realicen consultas de SELECT. Se pueden utilizar las mismas cláusulas que se utilizan en las consultas de selección tradiciones como ordenamientos (ORDER BY) y filtros (WHERE).
De ésta manera las vistas se transforman en una herramienta muy poderosa.

Código SQL y ejemplo práctico
Como podemos ver en el gráfico nº 2, de una estructura normalizada se puede obtener un conjunto de datos desnormalizado, simplificando la tarea de manipulación de datos por parte de quien utilice la vista.
A continuación veremos el código que permite generar la vista que obtiene los datos de los jugadores.
Para crear una vista se necesitará crear un objeto de la base de datos, el mismo deberá tener un nombre válido que no se repita y que no sea una palabra reservada. La sentencia utilizada para crear vistas es `CREATE VIEW` *`nombre_de_la_vista`* `AS`

Por ejemplo, si quisiera generar una vista que obtenga todos los jugadores debería escribir la siguiente consulta:

```
CREATE VIEW vw_ListadoJugadores AS SELECT * FROM JUGADORES
```

Aquí tenemos una vista que obtendrá los mismos datos que `SELECT * FROM JUGADORES`, vale aclarar que podremos aplicarle consultas de selección a nuestra vista permitiendo ejecutar, por ejemplo, `SELECT LJ.* FROM vw_ListadoJugadores LJ WHERE LJ.Apellido LIKE '%EZ'`

Veamos ahora un ejemplo más complejo, que utiliza una serie de subconsultas para obtener un resultado más estadístico de cada jugador.
Supongamos que deseamos obtener además del apellido y nombre, el equipo donde el jugador se desempeña, la cantidad de partidos jugados, la cantidad de tarjetas (rojas y amarillas), la cantidad de goles y el promedio de goles por partido. Para ello debemos incluir a la consulta principal un grupo de subconsultas.

```
SELECT J.APELLIDO, J.NOMBRE, E.EQUIPO,
(SELECT COUNT(IDPARTIDO) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS CANT_PARTIDOS,
(SELECT SUM(T_AMARILLA + T_ROJA) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS TOTAL_TARJETAS,
(SELECT SUM(GOLES_FAVOR) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS TOTAL_GOLES,
(SELECT AVG(GOLES_FAVOR * 1.00) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS PROMEDIO_GOLES
FROM JUGADORES J INNER JOIN EQUIPOS E ON E.IDEQUIPO = J.IDEQUIPO
```

	APELLIDO	NOMBRE	EQUIPO	CANT_PARTIDOS	TOTAL_TARJETAS	TOTAL_GOLES	PROMEDIO_GOLES
1	GABBARINI	ADRIAN	INDEPENDIENTE	1	1	0	0.000000
2	MILITO	GABRIEL	INDEPENDIENTE	1	3	0	0.000000
3	PARRA	FACUNDO	INDEPENDIENTE	1	1	2	2.000000
4	VEGA	DIEGO	RIVER PLATE	1	1	0	0.000000
5	DOMINGUIEZ	CHORI	RIVER PLATE	1	2	1	1.000000
6	TREZEGUET	DAVID	RIVER PLATE	2	3	3	1.500000

La consulta parecería ser muy difícil, pero se puede observar como a partir de la cuarta columna los valores provienen de subconsultas que se encuentran relacionadas a cada uno de los jugadores en cuestión.
Por ejemplo, supongamos que la primera fila es la del jugador 'Gabbarini' cuyo nombre es 'Adrian' y juega en 'Independiente' luego se desea obtener la cantidad de partidos jugados por lo que se realiza una subconsulta realizando un COUNT del campo IDPARTIDO de la tabla PARTIDOS_X_JUGADOR siempre y cuando el IDJUGADOR sea igual al IDJUGADOR de la tabla con el alias J que, para este registro, es el ID del jugador 'Gabbarini'.
La misma lógica se aplica a las siguientes columnas que se calculan mediante subconsultas y luego se aplicarán de la misma manera para los siguientes registros hasta llegar al último de los registros que corresponden al jugador 'David Trezeguet' de 'River'.
De ésta manera y gracias a la utilización de subconsultas podemos obtener en el mismo grupo de datos la información estadística de todos los jugadores. Pero, a medida que queramos incorporar filtros, ordenamientos, etc. La incorporación de más y más subconsultas incorporarán un nivel de complejidad demasiado alto a la consulta. Es por eso que para este tipo de casos resultan muy útiles las vistas.

Lo primero que haremos será transformar nuestra consulta estadística en una vista, para ello debemos ejecutar la siguiente consulta:

```
CREATE VIEW vw_EstadisticaJugadores
AS
SELECT J.APELLIDO, J.NOMBRE, E.EQUIPO,
(SELECT COUNT(IDPARTIDO) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS CANT_PARTIDOS,
(SELECT SUM(T_AMARILLA + T_ROJA) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS TOTAL_TARJETAS,
(SELECT SUM(GOLES_FAVOR) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS TOTAL_GOLES,
(SELECT AVG(GOLES_FAVOR * 1.00) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS PROMEDIO_GOLES
FROM JUGADORES J INNER JOIN EQUIPOS E ON E.IDEQUIPO = J.IDEQUIPO
```

Ahora sabemos que existe un objeto en nuestra base de datos que nos permitirá mediante un SELECT obtener los datos estadísticos de los jugadores. Un simple `SELECT * FROM vw_EstadisticaJugadores` nos abstraerá de la complejidad que implica obtener esa información. Y sobre todo, nos permitirá aplicar filtros y ordenamientos de manera que nuestro listado se transforme en otro más complejo.
De ésta manera obtener un listado de aquellos jugadores que hayan obtenido sólo una tarjeta y no hayan convertido ningún gol.

```
SELECT EJ.* FROM vw_EstadisticaJugadores EJ WHERE EJ.TOTAL_TARJETAS = 1 AND
TOTAL_GOLES = 0
```

	APELLIDO	NOMBRE	EQUIPO	CANT_PARTIDOS	TOTAL_TARJETAS	TOTAL_GOLES	PROMEDIO_GOLES
1	GABBARINI	ADRIAN	INDEPENDIENTE	1	1	0	0.000000
2	VEGA	DIEGO	RIVER PLATE	1	1	0	0.000000

Claramente, se puede notar como una consulta bastante compleja se simplifica enormemente utilizando subconsultas y vistas.
Pero supongamos que al ejecutar la siguiente consulta nos damos cuenta que nos falta algún dato importante, por ejemplo, aquí podríamos simular que nos falta el IDJUGADOR para que ésta consulta quede completa.
Para ello lo que debemos hacer es modificar la estructura de la vista utilizando la sentencia `ALTER VIEW` *`nombre_de_la_vista`* `AS`.
Para este caso en particular deberíamos ejecutar el ALTER VIEW de la vista vw_EstadisticaJugadores con la incorporación de la columna IDJUGADOR a la consulta de selección.

```
ALTER VIEW vw_EstadisticaJugadores
AS
SELECT J.IDJUGADOR, J.APELLIDO, J.NOMBRE, E.EQUIPO,
(SELECT COUNT(IDPARTIDO) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS CANT_PARTIDOS,
(SELECT SUM(T_AMARILLA + T_ROJA) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS TOTAL_TARJETAS,
(SELECT SUM(GOLES_FAVOR) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR = J.IDJUGADOR)
AS TOTAL_GOLES,
(SELECT AVG(GOLES_FAVOR * 1.00) FROM PARTIDOS_X_JUGADOR WHERE IDJUGADOR =
J.IDJUGADOR) AS PROMEDIO_GOLES
FROM JUGADORES J INNER JOIN EQUIPOS E ON E.IDEQUIPO = J.IDEQUIPO
```

Luego, cuando se ejecute la consulta `SELECT EJ.* FROM vw_EstadisticaJugadores EJ WHERE EJ.TOTAL_TARJETAS = 1 AND TOTAL_GOLES = 0` para obtener el listado que deseábamos podremos observar que al modificarse la estructura de la vista, nuestro *viejo* y erróneo listado pasa a actualizarse incorporando la columna que se necesitaba.

	IDJUGADOR	APELLIDO	NOMBRE	EQUIPO	CANT_PARTIDOS	TOTAL_TARJETAS	TOTAL_GOLES	PROMEDIO_GOLES
1	1	GABBARINI	ADRIAN	INDEPENDIENTE	1	1	0	0.000000
2	4	VEGA	DIEGO	RIVER PLATE	1	1	0	0.000000

Por último, es posible que deseemos que una vista no siga formando parte de nuestra base de datos. Para eliminar una vista de nuestra base lo que debemos ejecutar es la sentencia `DROP VIEW` *`nombre_de_la_vista`*.
En nuestro ejemplo se debería ejecutar `DROP VIEW vw_EstadisticaJugadores`