SUBCONSULTAS Y VISTAS

BASE DE DATOS

SUBCONSULTAS

- Una subconsulta en SQL consiste en utilizar los resultados de una consulta dentro de otra, que se considera la principal. Esta posibilidad fue la razón original para la palabra “estructurada” en el nombre Lenguaje de Consultas Estructuradas (Structured Query Language, SQL).

SUBCONSULTAS

```
SELECT dep_no “Nº Empleado”,apellido,salario
FROM empleados
WHERE salario>(SELECT AVG(salario)
FROM empleados );
```

La subconsulta (comando SELECT entre Paréntesis) se ejecuta primero y, posteriormente, el valor extraído es utilizado en la consulta principal

SUBCONSULTAS

donde el formato de la sentencia SELECT entre paréntesis tiene las siguientes diferencias

con la sentencia SELECT de las consultas:

- No tiene sentido la cláusula ORDER BY ya que los resultados de una subconsulta se utilizan internamente y no son visibles al usuario.
- Los nombres de columna que aparecen en una subconsulta pueden referirse a columnas de la tabla de la consulta principal y se conocen como referencias externas.

SUBCONSULTAS

- Las subconsultas se pueden usar en:

- En el listado de las columnas de una consulta SELECT.
- En una clausula WHERE, el ejemplo de arriba.
- En una clausula FROM.
- En una clausula FROM de una consulta UPDATE.
- En una clausula HAVING.}

- SELECT au_lname,
- au_fname,
- title FROM (SELECT au_lname, au_fname, au_id
- FROM pubs.dbo.authors
- WHERE state = 'CA') as A
- JOIN pubs.dbo.titleauthor ta ON A.au_id = ta.au_id
- JOIN pubs.dbo.titles t ON ta.title_id = t.title_id

- http://www.akadia.com/services/sqlsrv_subqueries.html

- http://www.databasejournal.com/features/mssql/article.php/3464481/Using-a-Subquery-in-a-T-SQL-Statement.htm

SUBCONSULTAS

Valores de retorno de las subconsultas y condiciones de selección.

Una subconsulta forma parte de la condición de selección en las cláusulas WHERE o HAVING

El resultado de una subconsulta puede ser un valor simple o más de un valor. Según el retorno de

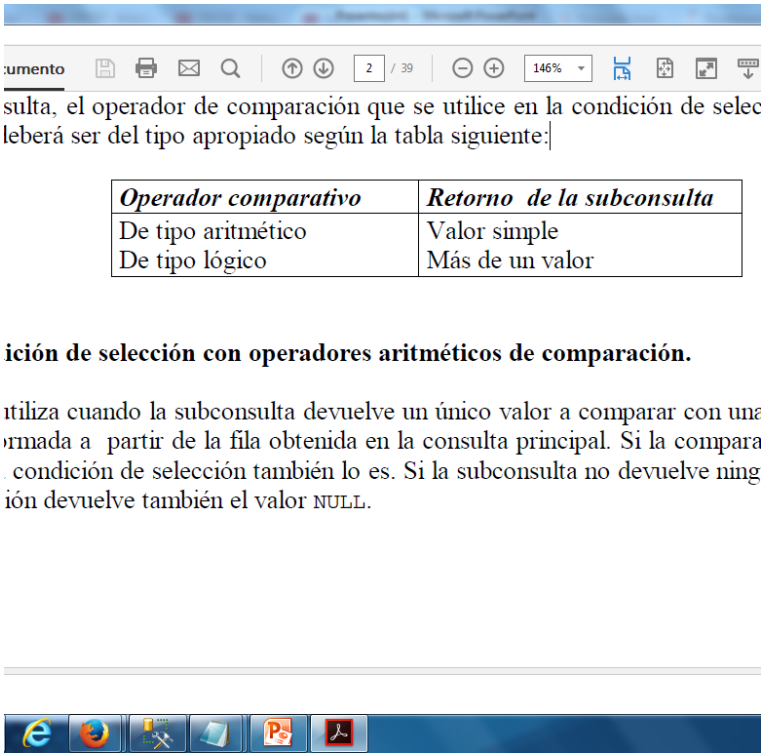
la subconsulta, el operador de comparación que se utilice en la condición de selección del WHERE o

HAVING deberá ser del tipo apropiado según la tabla siguiente:

SUBCONSULTAS

- **Condición de selección con operadores aritméticos de comparación**

- Se utiliza cuando la subconsulta devuelve un único valor a comparar con una expresión, por lo general formada a partir de la fila obtenida en la consulta principal. Si la comparación resulta cierta (TRUE), la condición de selección también lo es. Si la subconsulta no devuelve ninguna fila (NULL), la comparación devuelve también el valor NULL.



sulta, el operador de comparación que se utilice en la condición de selec
leberá ser del tipo apropiado según la tabla siguiente:

Operador comparativo	*Retorno de la subconsulta*
De tipo aritmético	Valor simple
De tipo lógico	Más de un valor

ición de selección con operadores aritméticos de comparación.

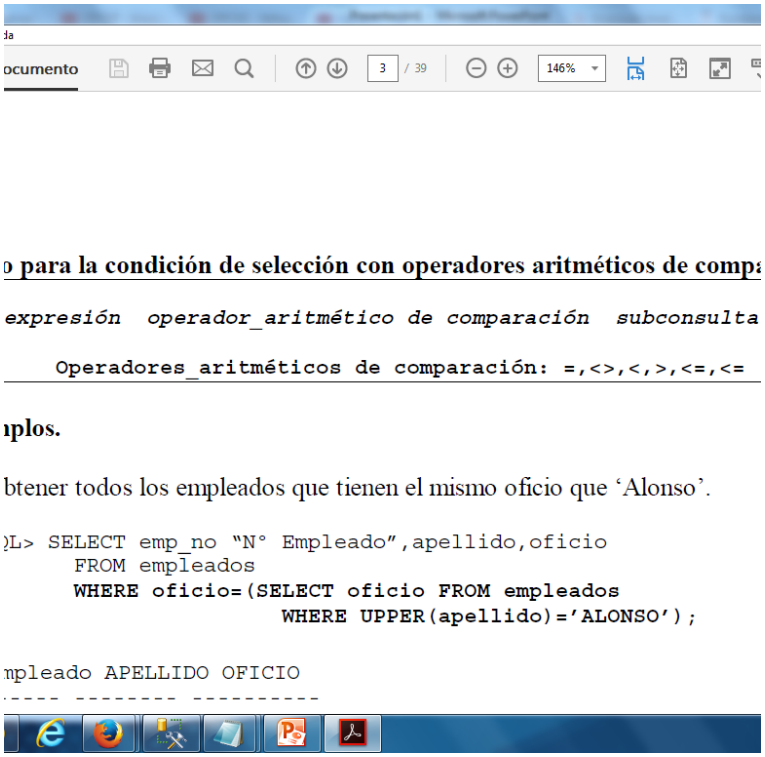
ıtiliza cuando la subconsulta devuelve un único valor a comparar con una
rmada a partir de la fila obtenida en la consulta principal. Si la compara
condición de selección también lo es. Si la subconsulta no devuelve ning
ión devuelve también el valor NULL.

SUBCONSULTAS

- Obtener todos los empleados que tienen el mismo oficio que ‘Alonso’.

- SELECT emp_no “Nº Empleado”,apellido,oficio
- FROM empleados
- WHERE oficio=(SELECT oficio FROM empleados
- WHERE UPPER(apellido)=’ALONSO’)

- Nº Empleado APELLIDO OFICIO
- ---------- -------- ----------
- 7499 ALONSO VENDEDOR
- 7654 MARTIN VENDEDOR
- 7844 CALVO VENDEDOR

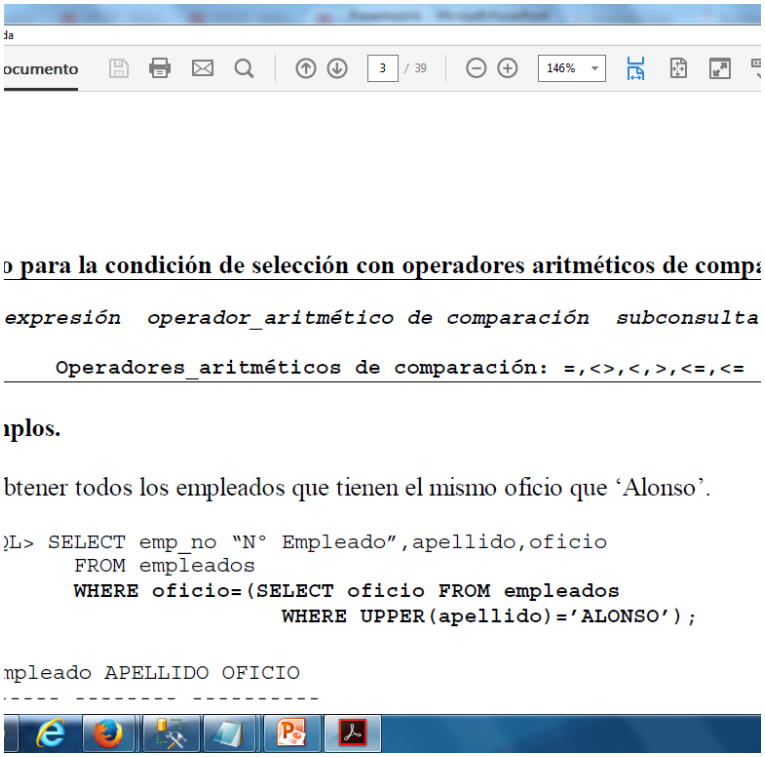


SUBCONSULTAS

- Obtener información de los empleados que ganan más que cualquier empleado del
- departamento 30.

- SELECT emp_no “Nº Empleado”,apellido,salario,dep_no
- “Nº Departamento”
- FROM empleados
- WHERE salario>(SELECT MAX(salario)
- FROM empleados
- WHERE dep_no=30)

- Nº Empleado APELLIDO SALARIO Nº Departamento
- ---------- -------- --------- -------------
-
- 7839 REY 600000 10



Condición de selección con operadores lógicos
SUBCONSULTAS

- Se utiliza cuando la subconsulta devuelve más de una fila a comparar con la fila actual de la consulta principal. Los operadores lógicos permitidos por la mayoría de los gestores de bases de datos son: IN, EXISTS, ANY y ALL. El más utilizado es el operador IN.
- **Operador lógico IN**
- Comprueba si valores de la fila actual de la consulta principal coincide con algunos de los devueltos por la subconsulta. Si el resultado es afirmativo la comparación resulta cierta (TRUE)

- Listar, en orden alfabético, aquellos empleados que no trabajen ni en Madrid ni en Barcelona

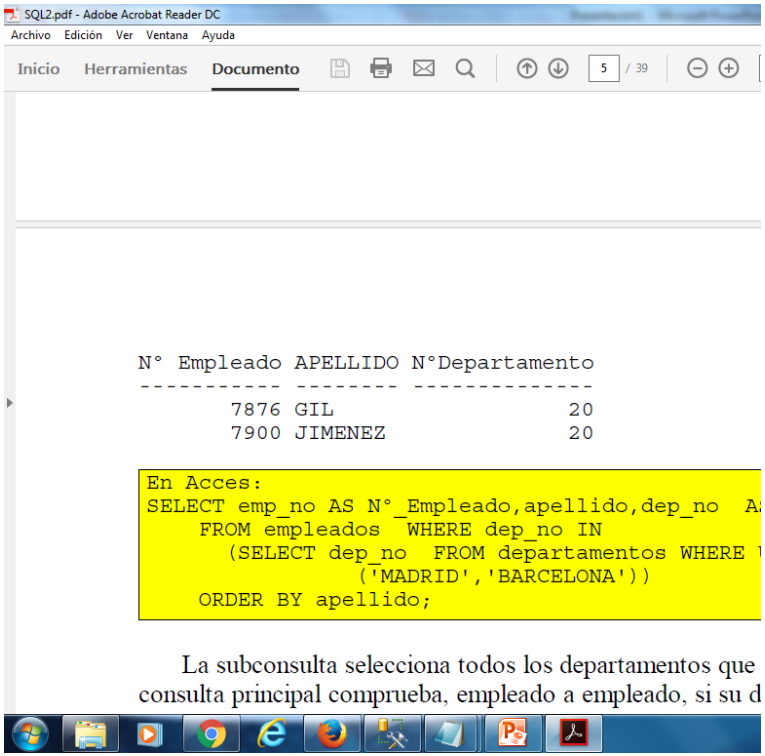
- SELECT emp_no “Nº Empleado”,apellido,dep_no
- “NºDepartamento”
- FROM empleados
- WHERE dep_no IN (SELECT dep_no
- FROM departamentos
- WHERE UPPER(localidad) NOT IN
- (‘MADRID’,’BARCELONA’))
- ORDER BY apellido;

Condición de selección con operadores lógicos
SUBCONSULTAS

- **Operador lógico EXISTS**

- Se utiliza cuando la condición de selección consiste exclusivamente en comprobar que la subconsulta devuelve alguna fila seleccionada según la condición incluida en la propia subconsulta.

- El operador EXISTS no necesita que la subconsulta devuelva alguna columna porque no utiliza ninguna expresión de comparación, justificando así la aceptación del * en el formato de la misma.



SUBCONSULTAS

- Una subconsulta expresada con el operador EXISTS también podrá expresarse con el operador IN.

- SELECT localidad
- FROM departamentos d
- WHERE EXISTS (SELECT *
- FROM empleados e
- WHERE comision>10*salario/100
- AND e.dep_no=d.dep_no);

- **Operadores lógicos ANY y ALL**

- Se utilizan junto a los operadores aritméticos de comparación para ampliar las posibles
- comprobaciones de valores obtenidos a partir de la fila seleccionada en la consulta principal con valores obtenidos en la subconsulta. Su uso a menudo es sustituido por el del operador **IN.**

SUBCONSULTAS

- Una subconsulta expresada con el operador EXISTS también podrá expresarse con el operador IN.

- SELECT localidad
- FROM departamentos d
- WHERE EXISTS (SELECT *
- FROM empleados e
- WHERE comision>10*salario/100
- AND e.dep_no=d.dep_no);

- **Operadores lógicos ANY y ALL**

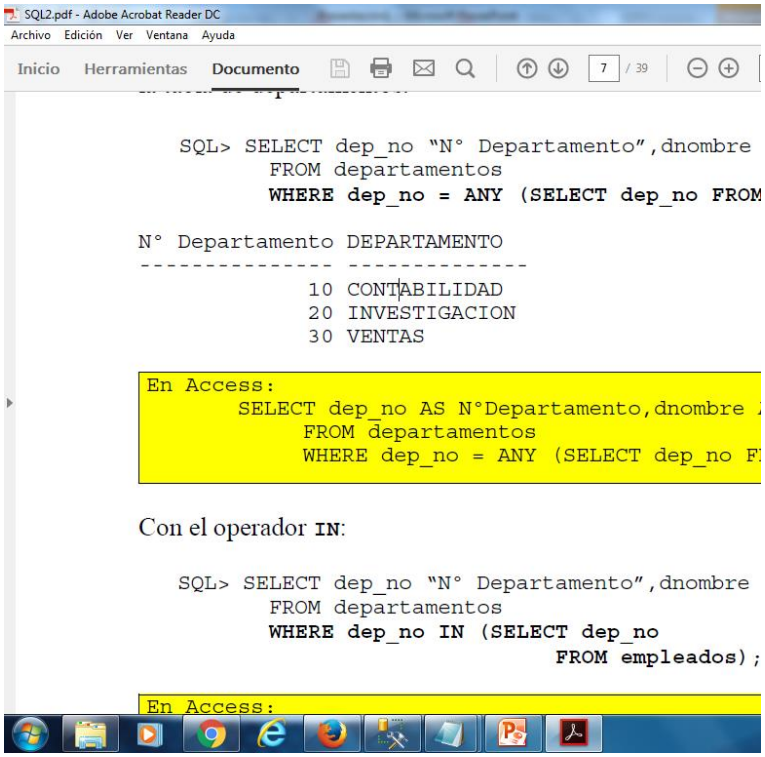
- Se utilizan junto a los operadores aritméticos de comparación para ampliar las posibles
- comprobaciones de valores obtenidos a partir de la fila seleccionada en la consulta principal con valores obtenidos en la subconsulta. Su uso a menudo es sustituido por el del operador **IN.**

SUBCONSULTAS

- El operador ANY con uno de los seis operadores aritméticos compara el valor de la expresión formada a partir de la consulta principal con valores producidos por la subconsulta. Si alguna de las comparaciones individuales produce un resultado verdadero (TRUE), el operador ANY devuelve un resultado verdadero(TRUE).

- Comprobar que todos los empleados tengan asignado un código de departamento existente en la tabla de departamentos.

- SELECT dep_no "Nº Departamento",dnombre Departamento
- FROM departamentos
- WHERE dep_no = ANY (SELECT dep_no FROM empleados)



SUBCONSULTAS

- SELECT dep_no ”Nº Departamento”,COUNT(*) “Total Empleados”
- FROM empleados
- GROUP BY dep_no
- HAVING COUNT(*)=(SELECT MAX(COUNT(*))
- FROM empleados
- GROUP BY dep_no);

SUBCONSULTAS

- **Subconsultas correlacionadas.**
- Cuando los nombres de columnas que aparecen en una subconsulta son nombres de columnas de la consulta principal o de otra subconsulta más externa, caso de las anidadas, se dice que son referencias externas y la subconsulta que es correlacionada.

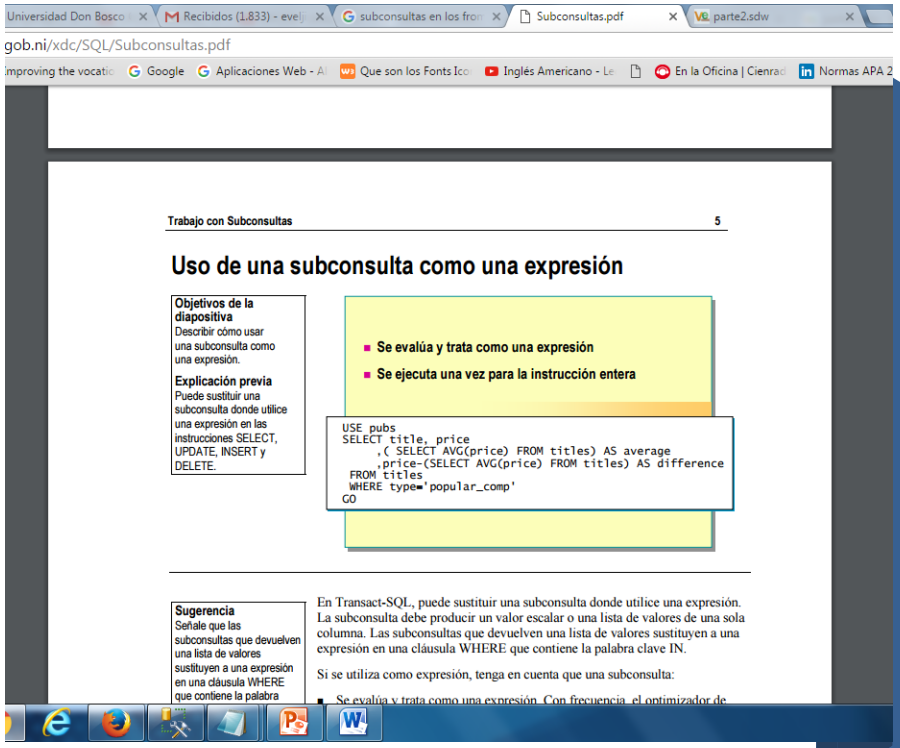
- Si en una subconsulta correlacionada coincide el nombre de una referencia externa con el nombre
- de alguna columna de la tabla que está siendo seleccionada en la subconsulta, se deberá asignar un alias a cada tabla y se utilizará para identificar cada columna.
- Visualizar el número de departamento, el oficio y el salario de los oficios con mayor salario de cada departamento.

- SELECT dep_no “Nº Departamento”,oficio,salario
- FROM empleados e1
- WHERE salario=(SELECT MAX(salario)
- FROM empleados e2
- **WHERE e1.dep_no=e2.dep_no**)

SUBCONSULTAS

- **Subconsultas correlacionadas.**
- Nº Departamento OFICIO SALARIO
- ---------- ---------- ---------
- 30 DIRECTOR 385000
- 10 PRESIDENTE 600000
- 20 ANALISTA 335000

- La ***referencia externa*** sería **dep_no** y, aunque sea a la misma tabla, la subconsulta la trata como
- otra tabla, de la que la suya es copia. Asignamos alias para distinguir la referencia dep_no a una u
- otra tabla.



SUBCONSULTAS

SUBCONSULTAS

- En este ejemplo se devuelve el precio de un popular libro de informática, el
- promedio del precio de todos los libros y la diferencia entre el precio del libro y
- el promedio del precio de todos los libros.

```
USE pubs
go
SELECT title, price
 ,(SELECT AVG(price) FROM titles) AS average
 ,price-(SELECT AVG(price) FROM titles) AS difference
 FROM titles
 WHERE type='popular_comp'
GO
```

SUBCONSULTAS

```
-- Peliculas dirigidas por
SELECT P.nombrePelicula as "Nombre Pelicula",
 P.year as Año,
 R.nombreRol as Rol,
        Actor
FROM Pelicula P,
 Rol R,
 Actua D,
       (SELECTA.idActor, A.nombreActor Actor,
       N.nombreNacionalidad Nacionalidad
        FROMActor A, Nacionalidades N
        WHERE   N.idNacionalidad = A.idNacionalidad
        AND     N.nombreNacionalidad = 'Mexicana') A
WHERE    D.idRol = R.idRol
AND D.idPelicula = P.idPelicula
AND      (R.nombreRol = 'Director')
AND      (A.idActor = D.idActor);
```

Vistas

- La vista almacena una consulta como un objeto para utilizarse posteriormente. Las tablas consultadas en una vista se llaman tablas base. En general, se puede dar un nombre a cualquier consulta y almacenarla como una vista.
- La vista suele llamarse también tabla virtual porque los resultados que retorna y la manera de referenciarlas es la misma que para una tabla.

Vistas

Las vistas permiten:

- Ocultar información: permitiendo el acceso a algunos datos y manteniendo oculto el resto de la información que no se incluye en la vista. El usuario opera con los datos de una vista como si se tratara de una tabla, pudiendo modificar tales datos.
- Simplificar la administración de los permisos de usuario: se pueden dar al usuario permisos para que solamente pueda acceder a los datos a través de vistas, en lugar de concederle permisos para acceder a ciertos campos, así se protegen las tablas base de cambios en su estructura.
- Mejorar el rendimiento: se puede evitar tipear instrucciones repetidamente almacenando en una vista el resultado de una consulta compleja que incluya información de varias tablas.

Vistas

- En el siguiente ejemplo creamos la vista "vista_empleados", que es resultado de una combinación en la cual se muestran 4 campos:

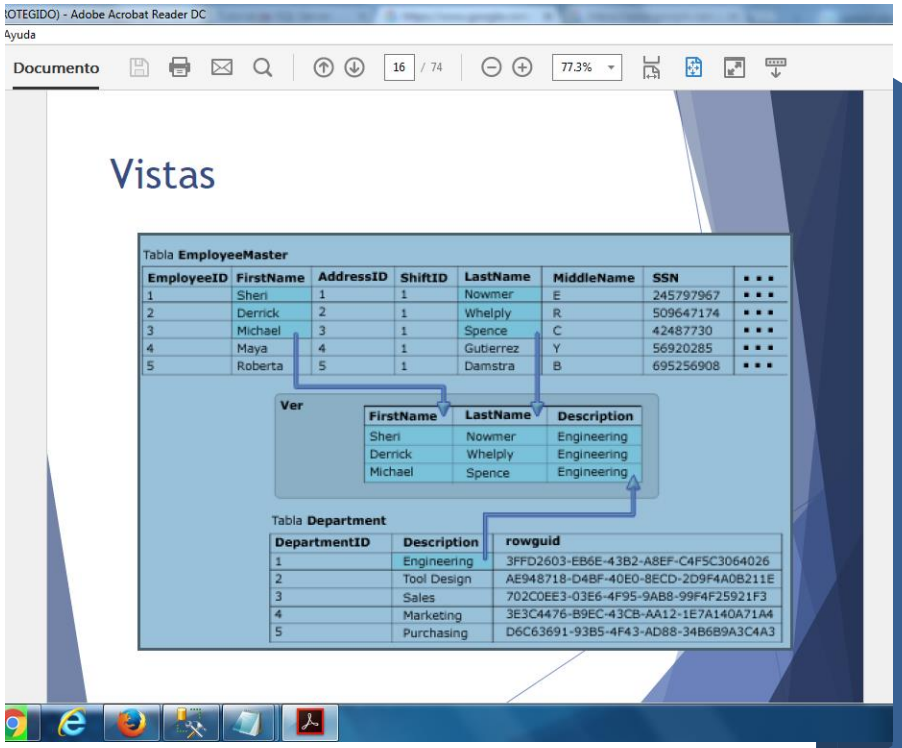
```
create view vista_empleados as
 select (apellido+' '+e.nombre) as nombre,sexo,
  s.nombre as seccion, cantidadhijos
  from empleados as e
  join secciones as s
  on codigo=seccion
```

- Para ver la información contenida en la vista creada anteriormente tipeamos:

```
select *from vista_empleados;
```

- Podemos realizar consultas a una vista como si se tratara de una tabla:

```
select seccion,count(*) as cantidad
 from vista_empleados;
```


(OTEGIDO) - Adobe Acrobat Reader DC
Ayuda
Documento
16 / 74
77.3%
Vistas
Tabla EmployeeMaster
EmployeeID FirstName AddressID ShiftID LastName MiddleName SSN
1 Sheri 1 1 Nowmer E 245797967
2 Derrick 2 1 Whelply R 509647174
3 Michael 3 1 Spence C 42487730
4 Maya 4 1 Gutierrez Y 56920285
5 Roberta 5 1 Damstra B 695256908
Ver
FirstName LastName Description
Sheri Nowmer Engineering
Derrick Whelply Engineering
Michael Spence Engineering
Tabla Department
DepartmentID Description rowguid
1 Engineering 3FFD2603-EB6E-43B2-A8EF-C4F5C3064026
2 Tool Design AE948718-D4BF-40E0-8ECD-2D9F4A0B211E
3 Sales 702C0EE3-03E6-4F95-9AB8-99F4F25921F3
4 Marketing 3E3C4476-B9EC-43CB-AA12-1E7A140A71A4
5 Purchasing D6C63691-93B5-4F43-AD88-34B6B9A3C4A3

Vistas

Procedimientos Almacenados

- Microsoft SQL Server provee el mecanismo de procedimientos almacenados para simplificar el porceso de desarrollo de la base de datos agrupando sentencias de Transact-SQL en bloques manejables.

Procedimientos Almacenados Beneficios

Por qué utilizar procedimientos almacenados?

- Ejecución Precompilada. SQL Server compila cada procedimiento almacenado una vez y entonces reutiliza el plan de ejecución. Esto resulta en una tremenda mejora de rendimiento cuando estos procedimientos son llamados repetidamente.

Procedimientos Almacenados Beneficios

- Tráfico cliente/servidor reducido. Si el ancho de banda de la red es una preocupación. los procedimientos almacenados pueden reducir largas consultas SQL en una sola línea siendo transmitida a lo largo de los cables.

Procedimientos Almacenados Beneficios

- Reuso eficiente de código y abstracción de la programación. Los procedimientos almacenados pueden ser usados por múltiples usuarios y programas cliente. Si se utilizan de manera planificada, se encontrará que el ciclo de desarrollo toma menos tiempo.

Procedimientos Almacenados Beneficios

- Mejora en los controles de seguridad. Se puede conceder permiso a los usuarios para ejecutar un procedimiento almacenado independientemente de los permisos de la tabla dependiente.

Procedimientos Almacenados Estructura

- Los procedimientos almacenados son extremadamente similares a los elementos vistos en otros lenguajes de programación. Aceptan datos en forma de parámetros de entrada que son especificados en tiempo de ejecución.

Procedimientos Almacenados Estructura

- Estos parámetros de entrada (si son implementados) son utilizados en la ejecución de una serie de sentencias que producen algún resultado. Este resultado es devuelto al ambiente que lo está llamando a través del uso de un recordset, parámetros de salida y código de devolución.

Procedimientos Almacenados Estructura

```
CREATE PROCEDURE num_productos @productos int OUTPUT AS
SELECT * FROM Products
SELECT @productos = @@ROWCOUNT
RETURN (0)
```

Procedimientos Almacenados Estructura

```
DECLARE @productos int
EXEC num_productos @productos OUTPUT
SELECT [Total de productos] = @productos
```