

Tema 5.

SQL en lenguajes de programación

Índice

- La cadena de conexión con la BD
- SQL en ASP.NET
 - Conocimientos básicos
 - Controles de servidor web: básicos, enlazados a datos y de origen de datos
- Codificación de una instrucción SQL
 - Pasos
 - Ejemplos en VB.NET y OLEDB: código videoclub web
- Transacción
 - Definición y contexto
 - Propiedades ACID
 - Codificación

La cadena de conexión con la BD

- Para poder programar una aplicación que utilice una base de datos, debe estar instalado el **conector** adecuado (driver), que depende de la tecnología de programación (lenguaje y forma de conexión), del SO y del SGBD. Este driver dará acceso a todos los métodos necesarios para poder trabajar con ese tipo de base de datos en el entorno elegido.
- **Driver para Access Java JDBC** <http://ucanaccess.sourceforge.net/site.html>
- **Drivers para MySQL** <https://dev.mysql.com/downloads/connector/>
- Una **cadena de conexión** es un dato de tipo cadena que contiene los datos para conectarse a la base de datos a utilizar. Ejemplos:
 - para **MS Access 2007 o superior** (utiliza **tecnología OLEDB**)
“Provider=Microsoft.ACE.OLEDB.12.0; Data Source=ruta\nombreBD.accdb”
 - para **MySQL** (con **tecnología ODBC**)
“DRIVER={MySQL ODBC 5.1 Driver} ; SERVER=127.0.0.1;
DATABASE=nombreBD; USER=usuario; PASSWORD=contraseña; Option=3”
 - **Info cadenas de conexion en** <https://www.connectionstrings.com/>

SQL en ASP.NET: Conocimientos básicos

- ASP.NET: Marco de trabajo / tecnología de Microsoft para programación de aplicaciones Web (sitios y páginas Web).
- Carpetas y ficheros destacables en una aplicación Web ASP.NET:
 - **Account:** Carpeta que contiene todo lo relacionado con el login.
 - **App_Data:** Directorio por defecto para las bases de datos. Contiene la base de datos ASPNETDBXXX.mdf (de MS SQL Server) que guarda los datos de los usuarios registrados en la aplicación Web.
 - **Site.Master:** Página maestra. Define la apariencia de las páginas del sitio: Estilo común para todo el Sitio (colores, dimensiones de la página...), opciones de Menú, etc.
 - **Default.aspx:** Es la página web por defecto, se crea automáticamente.
 - **Web.Config:** Contiene cadenas de conexión, autorizaciones, etc.
- Las páginas Web se denominan **formularios** (Form Web).
 - El diseño de cada formulario Web (lo que se representa en el navegador, la interfaz de usuario) se almacena en un fichero con extensión **.aspx**.
 - La programación asociada a cada formulario se realiza en VB.NET (o en C#) y se almacena con el mismo nombre que el del formulario y extensión **.aspx.vb** (o .aspx.cs).

FUNCIONES ADMINISTRADOR

Se hereda de la página Maestra
(del fichero Site.Master)

Realice todas las funciones que quiera . Rellene los datos oportunos y pulse el botón.

Control Etiqueta o simple literal

Cambiar estado de un usuario

Cuenta del usuario a cambiar de estado

<Visualizar aquí los datos del usuario>

Control Caja de texto

Cambiar estado

Mostrar datos usuario

Elementos en
el **diseño** de
un formulario
Web

Dar de alta una película

Código de la película a dar de alta

<Poner aquí el resto de datos a pedir>

Control Botón

Dar de alta película

Dar de baja una película

Código de la película a dar de baja

<Visualizar aquí los datos de la peli>

Dar de baja esta película

Mostrar datos película

[Volver a la Página Principal](#)

SQL en ASP.NET: Controles de servidor Web

- En el diseño de sus páginas Web, ASP.NET utiliza **controles de servidor web**. Todo control de un formulario tiene su propio identificador ID.
- Algunos **tipos** de controles (disponibles en el cuadro de Herramientas):
 - **Básicos**: Label, TextBox y Button.
 - **Enlazados a datos**: DropDownList, DataList y GridView.
 - **De origen de datos**: Asocian los datos de una BD con los controles enlazados a datos, a través de una SELECT a esa BD.
- Cada tipo de control tiene además:
 - **propiedades** (con valores que se pueden leer y/o modificar) y
 - **eventos** asociados (que se pueden programar).

FUNCIONES USUARIO REGISTRADO

Realice todas las funciones que quiera . Rellene los datos oportunos y pulse el botón.

Modificar mis datos personales

Nombre y apellidos

Conchi Presedo

Modificar datos

Dirección

Manuel Moreno Pitxitxi, 3

Aumentar mi crédito

Introduzca aquí la cantidad en euros

TextBox

Button

Aumentar crédito

Alquilar película

Seleccione de la lista la película a alquilar

<insertar aquí
DropDownList>

Alquilar

Devolver película

Seleccione de la lista la película a devolver

<insertar aquí
DropDownList con
los nombres de las
pelis>

Devolver

Volver a la Página Principal

Los dos controles recuadrados a la derecha son los involucrados en la funcionalidad para aumentar el crédito del usuario registrado

CATÁLOGO PELÍCULAS

GridView

[titulo](#)

Star Wars: Episode IV - A New Hope (1977)

Finding Nemo (2003)

Forrest Gump (1994)

American Beauty (1999)

Pirates of the Caribbean: The Curse of the Black P

1 [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) ...

Label

ESTRENOS QUINCENA (fecha de hoy= 20/03/2023 8:58:11)

titulo

fecha_alta

Mamma Mia: Here We Go Again! (2018) 15/03/2023 7:36:10

Tag (2018) 10/03/2023 7:36:10

BlacKkKlansman (2018) 07/03/2023 7:36:10

GridView

FUNCIONES ADMINISTRADOR

Realice todas las funciones que quiera . Rellene los datos oportunos y pulse el botón.

Cambiar estado de un usuario

Cuenta del usuario a cambiar de estado

DataList

Nombre y apellido: Pepe Lopetegui

Dirección: Rue del Percebe, 13

Crédito: 0

Fecha y hora del alta: 05/03/2023 13:32:46

Estado: A

DropDownList

pepe ▼
gorka
miren
pepe
pepe2
vane

Cambiar estado

Mostrar/Ocultar usuario

SQL en ASP.NET: Controles de servidor Web (2)

- Ejemplo de programación de un **control básico**:
 - Control: Botón **Aumentar Crédito** (**ID: Aumentar**)
 - Tipo de control: control básico Button
 - Funcionalidad: Al hacer clic en el botón Aumentar Crédito se debe sumar al crédito del usuario el que haya en la caja de texto.
 - Evento a programar del control: click
 - Código asociado a ese evento:
 - Modificar el crédito (instrucción SQL: UPDATE)
 - Otros datos necesarios:
 - El valor que contiene el TextBox que aparece a su izquierda (**ID: cantidadEuros**). Este valor se guarda en la propiedad Text del control.
 - El valor del usuario que ha iniciado la sesión.

SQL en ASP.NET: Controles de servidor Web (3)

- Existen dos tipos de **controles de servidor web** para vincular datos de una BD a la aplicación Web.
- **Tipos** de controles:
 - Los controles **de origen de datos** (objetos **XXXDataSource**), que administran las tareas de conexión a un origen de datos y de lectura y escritura de datos. Actúan como intermediarios entre la BD y los demás controles de la página Web ASP.NET.
 - Los controles **enlazados a datos** (por ejemplo los objetos **DropDownList**, **GridView** y **DataList**), que representan los datos en el navegador. Un control enlazado a datos se vincula - a través de su propiedad `DataSourceID` - a un control de origen de datos y recupera los datos automáticamente en el momento apropiado.
- Permiten diseñar y realizar las operaciones de datos más comunes con una BD sin necesidad de escribir una sola línea de código VB.NET.
- **En VB.NET:** los datos de la BD se pueden cargar/refrescar mediante ***NombreControl.DataBind()*** donde *NombreControl* es el ID del control enlazado a datos.

SQL en ASP.NET: Controles de servidor Web (4)

- En el **diseño de un control enlazado a datos** se dan 4 pasos consecutivos, que consisten en crear y configurar tanto el control enlazado a datos como su origen de datos asociado.
- Pasos:
 - Paso 1. **Crear el elemento de control con enlace a datos** arrastrándolo desde el cuadro de Herramientas.
 - Paso 2. **Crear un nuevo control de origen de datos para el elemento** anterior, si no está creado ya. Se crea la cadena de conexión con la BD , que se guarda en el fichero **Web.config** de la aplicación Web.
 - Paso 3. **Configurar el control de origen de datos.** Se prepara la SELECT que se ejecutará utilizando la conexión a la BD creada en el paso 2.
 - Paso 4 (Opcional). **Configurar el elemento de control con enlace a datos.** Se puede dar un formato a los datos y/o habilitar más funcionalidades.
- Estos 4 pasos se hacen con los asistentes de los controles, disponibles en la **vista Diseño** del Formulario Web.

SQL en ASP.NET: Controles de servidor Web (5)

- En el **Paso 3 del diseño** del control enlazado a datos, la SELECT del origen de datos puede aceptar parámetros (veremos cómo en la pág. 15). Sus valores pueden tener diversos orígenes, lo que determina su tipo.
- **Tipos de parámetros** más comunes:
 - **Parámetro de sesión** (objeto **SessionParameter**): Establece un parámetro con el valor de una variable de sesión.
 - Una **variable de sesión** guarda el valor de una variable durante el tiempo que el usuario visita el sitio Web. Se declara en el código **VB.NET** (normalmente en `_Default.aspx.vb`) y se le da un valor con la instrucción:
Session("nombredelaVariabledeSesión") = valor.
 - **Parámetro de control** (objeto **ControlParameter**): Establece un parámetro con el valor de propiedad de un control de servidor de una página Web ASP.NET. (Ejemplos : Text y SelectedValue).
- Si el dato proviene de un control diseñado en el mismo formulario Web que la instrucción SQL es un parámetro de control. En caso contrario, se codifica como parámetro de sesión.

Codificación de una instrucción SQL

Para implementar una instrucción SQL en un lenguaje de programación para una aplicación se deben dar cuatro **pasos** consecutivos:

Paso 1. Crear y abrir una conexión con la BD

El ordenador debe tener instalado previamente el conector/driver correspondiente al entorno de programación a utilizar y el usuario de ese ordenador debe tener acceso a él.

Paso 2. Preparar la instrucción SQL

Paso 3. Ejecutar la instrucción SQL

Paso 4. Cerrar la conexión y liberar memoria

- **Ejemplos de código** VB.NET con MS ACCESS en _Default.aspx.vb y FormularioFuncionesUsuario.aspx.vb (Págs de 14 a 18)

Código VB.NET en la aplicación Web:

Variables

- Los métodos que proporciona el driver para OLEDB (indicado en Provider) y que permiten trabajar con MS Access, están en el espacio de nombres / paquete **System.Data.OleDb** (línea 1)
- Variable **usuario** (línea 6):
Cadena que contiene el nombre del usuario que ha iniciado la sesión en la web de la plataforma de streaming. **Variable de control local** , es decir, solamente válida en la página web en la que está declarada.
- Variable **cadenaconexion** (línea 8):
Variable de tipo String que guarda la cadena de conexión con la BD.
- Variable de sesión **usuarioLogin** (línea 12):
También contiene el nombre del usuario. **Variable de sesión**, es decir, *válida durante toda la sesión del usuario logueado*, desde entrar hasta salir, lo que puede implicar varias páginas web. Puede ser necesaria para programar el origen de datos de algunos elementos de control en ASP.NET (como el GridView).

```
1 Imports System.Data.OleDb
2
3 Public Class _Default
4     Inherits System.Web.UI.Page
5     'Asigna a Usuario el LoginName actual pasado a minúsculas (para las comparaciones)
6     Dim usuario As String = StrConv(System.Web.HttpContext.Current.User.Identity.Name, VbStrConv.Lowercase)
7     'Indicamos la cadena de conexion (tipo OLEDB)
8     Dim cadenaConexion As String = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\TEMP\Plataforma.accdb"
9
10    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
11        'Crea una VARIABLE DE SESIÓN llamada usuarioLogin y le asigna el LoginName actual pasado a minúsculas
12        Session("usuarioLogin") = StrConv(System.Web.HttpContext.Current.User.Identity.Name, VbStrConv.Lowercase)
13        If Page.IsPostBack = False And usuario <> "" Then 'Solamente se hace cuando vaya a esta página
14            '*****
15            ' CUANDO SE REGISTRA UN NUEVO USUARIO SE LE DA AUTOMÁTICAMENTE DE ALTA COMO USUARIO EN LA BD
16            '*****
17            'PASO 1. ABRIR LA CONEXION VERIFICANDO QUE SEA CORRECTA.
18            'Declarar y crear una nueva conexión
19            Dim conexion As OleDb.OleDbConnection
20            conexion = New OleDb.OleDbConnection(cadenaConexion)
21            'Intentar abrirla
22            Try
23                If conexion.State = ConnectionState.Closed Then
24                    conexion.Open()
25                End If
26            Catch ex As Exception
27                'No se ha podido abrir la conexión: Se debe repasar la cadena de conexión
28                MsgBox("Error al conectar con la base de datos")
29                Response.Write(ex.Message)
30                Response.Write(". Esta es tu cadena de conexion: " & cadenaConexion)
31                Response.End()
32            End Try
33        End If
15
```

```

34 'PASO 2. PREPARAR LA INSTRUCCION SQL A EJECUTAR
35 'Insertamos el usuario en la tabla USUARIOREG si no existe --> EJEMPLO DE INSTRUCCIÓN CON UN PARÁMETRO
36 Dim strSQL As String = "INSERT INTO USUARIOREG(usuarioLogin) VALUES (?)"
37 Dim dbComm As New OleDbCommand(strSQL, conexion)
38 dbComm.Parameters.Add("usuario", OleDbType.VarChar, 50)
39 dbComm.Parameters("usuario").Value = usuario
40
41 'PASO 3. EJECUTAR LA INSTRUCCION VERIFICANDO QUE SEA CORRECTA (depende de si es una SELECT o no)
42 Try
43     'Ejecutamos el INSERT
44     dbComm.ExecuteNonQuery() 'Ejecuta cualquier instrucción que no sea SELECT
45     MsgBox("Se ha dado de alta a un nuevo usuario registrado")
46 Catch exSql As Exception
47     'Si da error es que el usuario ya existe. En realidad no es un error para nosotros.
48     'No hacemos nada
49 Finally
50     'PASO 4. CERRAR conexion Y LIBERAR MEMORIA
51     'Cerramos la conexión y liberamos la memoria que hayamos podido utilizar
52     If conexion.State = ConnectionState.Open Then
53         conexion.Close()
54         conexion.Dispose()
55     End If
56 End Try
57 End If
58 End Sub
59
60 End Class

```



```

1 Imports System.Data.OleDb 'Para las conexiones tipo OleDb -- ACCESS
2
3 Public Class FormularioFuncionesUSUARIO
4     Inherits System.Web.UI.Page
5     'Asigna a Usuario el LoginName actual pasado a minúsculas (para las comparaciones)
6     Dim usuario As String = StrConv(System.Web.HttpContext.Current.User.Identity.Name, VbStrConv.Lowercase)
7     'Indicamos la cadena de conexion (tipo OLEDB)
8     Dim cadenaConexion As String = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\TEMP\Plataforma.accdb"
9
10 Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
11     'Comprobamos que se haya iniciado sesión
12     If usuario = "" Then
13         MsgBox("Debes Iniciar Sesión como usuario registrado para poder operar aquí")
14         Response.Redirect("default.aspx")
15     End If
16     If Page.IsPostBack = False Then 'Solamente se hace cuando vaya a esta página
17         'DECLARAR LAS VARIABLES NECESARIAS para las instrucciones de BD
18         Dim conexion As OleDb.OleDbConnection
19         Dim strSQL As String
20         Dim dbComm As OleDbCommand
21         'PASO 1. CREAR UNA CONEXION Y ABRIRLA
22         conexion = New OleDb.OleDbConnection(cadenaConexion)
23         conexion.Open()
24         '*****
25         ' SE RECUPERA EL ESTADO DEL USUARIO QUE HA INICIADO LA SESIÓN
26         '*****
27         'PASOS 2 Y 3. PREPARAR LA INSTRUCCION SQL Y EJECUTARLA
28         strSQL = "SELECT estado FROM USUARIOREG WHERE usuarioLogin=?"
29         dbComm = New OleDbCommand(strSQL, conexion)
30         dbComm.Parameters.Add(New OleDbParameter("usuario", OleDbType.VarChar)).Value = usuario
31         Dim estado As String 'Para guardar el estado del usuario en la aplicación
32         estado = dbComm.ExecuteScalar() ' Ejecuta una SELECT en la que solo se obtiene un dato
33

```

Caso 2
(Pasos
2 y 3)

```

34      '*****
35      ' SI ES UN USUARIO REGISTRADO ACTIVO (su estado es A), SE MUESTRAN SUS DATOS PERSONALES
36      '*****
37      If estado <> "A" Then
38          'Si su estado no es activo visualizar un mensaje
39          MsgBox("Has sido dado de baja por el administrador de la plataforma. No puedes operar.")
40          Response.Redirect("default.aspx")
41      Else
42          'Recuperar los demás datos del usuario desde la BD y mostrarlos -- EJEMPLO DE SELECT
43          'PASO 2. Preparar la instrucción SQL a ejecutar
44          strSQL = "SELECT nombre_apellido,direccion FROM USUARIOREG WHERE usuarioLogin='" & usuario & "'"
45          dbComm = New OleDbCommand(strSQL, conexion)
46          'PASO 3. Ejecutarla
47          Dim datosUsuarioReader As OleDbDataReader
48          datosUsuarioReader = dbComm.ExecuteReader() 'Ejecuta una SELECT que obtiene varios datos
49          'Tratar el resultado, es decir, los datos obtenidos por la select
50          While datosUsuarioReader.Read() 'Si hay varias filas las va leyendo una por una
51              'Se asignan los datos recuperados a los distintos TextBox de la página Web
52              Me.Nombre.Text = datosUsuarioReader(0) 'Primer dato de la fila
53              Me.Direccion.Text = datosUsuarioReader(1) 'Segundo dato de la fila
54          End While
55      End If
56      'PASO 4. CERRAR LA CONEXIÓN Y LIBERAR MEMORIA
57      conexion.Close()
58      conexion.Dispose()
59  End If
60 End Sub

61
62
63 Protected Sub VolverPrincipal_Click(ByVal sender As Object, ByVal e As EventArgs) Handles VolverPrincipal.Click
64     Response.Redirect("default.aspx")
65 End Sub
66 End Class

```

Caso 3 (Pasos 2 y 3)

Codificación de una instrucción SQL (2)

Paso 1. Crear y abrir una conexión con la BD

- Declarar variable para la conexión (si no está declarada ya)
`Dim conexion As OleDbConnection`
- Instanciar una conexión y abrirla (si no está ya)
`conexion=New OleDb.OleDbConnection(cadena de conexión)`
`conexion.Open()`
- Ejemplos líneas código: - (Default) de 18 a 33 - (Usuario) de 21 a 23

Paso 4. Cerrar la conexión y liberar memoria

- Cerrar la conexión : `conexion.Close()`
- Liberar memoria: `conexion.Dispose()`
- Ejemplos líneas código: - (Default) de 50 a 55 - (Usuario) de 56 a 58

Codificación de una instrucción SQL (3)

Paso 2. Preparar la instrucción SQL

- Declarar variables (si no están declaradas ya)

```
Dim strSQL As String 'cadena para la instrucción SQL
```

```
Dim dbComm As OleDbCommand 'comando listo para ejecutar
```

- Asignar la instrucción SQL a la cadena e instanciar un nuevo comando con esa cadena y la conexión (del paso 1)

```
strSQL="Instrucción SQL con una interrogación (?) en cada sitio  
donde vaya un parámetro"
```

```
dbComm=New OleDbCommand(strSQL,conexion)
```

- Añadir **por orden** un nombre y un valor para cada (?) que aparezca en la cadena.
Para cada parámetro:

```
dbComm.Parameters.Add("nombreParametro",OleDbType.TipoSQL))
```

```
dbComm.Parameters("nombreParametro").Value = valor
```

- Ejemplos líneas código: - (Default) de 34 a 39 - (Usuario) de 28 a 30
- (Usuario) de 42 a 45 (sin declarar parámetros NO RECOMENDADO)

Codificación de una instrucción SQL (4)

Paso 3. Ejecutar la instrucción SQL

- Para ejecutarla hay que utilizar un método del comando (variable dbComm del Paso 2), que depende de la instrucción SQL y del resultado que se va a obtener. Tres casos:

Caso 1. Cualquier instrucción que NO sea una SELECT

`dbComm.ExecuteNonQuery()`

- Ejemplo líneas código: - (Default) de 42 (empieza en la 37) a 46

Caso 2. SELECT en la que se obtiene un único valor

- Declarar una variable que guarde el resultado de la SELECT
`Dim var as TipoDelResultadoQueObtiene`
- Ejecutar la SELECT y guardar el resultado en esa variable
`var = dbComm.ExecuteScalar()`
- Ejemplo líneas código: - (Usuario) de 27 a 32

Codificación de una instrucción SQL (5)

Caso 3. Ejecutar cualquier otra SELECT (varios valores)

- Declarar una variable que guarde el resultado de la instrucción

`Dim resultado As OleDbDataReader`

- Ejecutar la SELECT y guardar el resultado en esa variable

`resultado = dbComm.ExecuteReader()`

- Tratar el resultado guardado, fila por fila mediante un bucle

`While resultado.Read()` 'lee la siguiente fila mientras haya

Aquí se trata la fila leída teniendo en cuenta que los datos se referencian como `resultado(0)` primer dato de esa fila, `resultado(1)` segundo dato de esa fila, `resultado(2)` tercer dato de esa fila, etc.

`End While`

- Ejemplos líneas código: - (Usuario) de 42 a 54

Codificación de una instrucción SQL (6)

Ejemplo:

- Preparar y ejecutar la instrucción SQL del botón “**Aumentar Crédito**”:
UPDATE USUARIOREG SET CREDITO =CREDITO + valorDeLaCajaDeTexto
WHERE usuarioLogin= usuarioQueHaIniciadoLaSesion

Otras tareas:

Páginas 24 y 25 de este mismo tema: Funciones administrador y funciones usuario registrado de la plataforma.

Pensar en las instrucciones SQL que deberán ir en cada uno de los botones (cada instrucción y sus parámetros) , junto con el tipo de método (caso 1 , 2 ó 3) necesario para ejecutar cada una de ellas.

FUNCIONES ADMINISTRADOR

Realice todas las funciones que quiera . Rellene los datos oportunos y pulse el botón.

**Hacerlos con
DropDownList**

Cambiar estado de un usuario

Cuenta del usuario a cambiar de estado

<Visualizar aquí los datos del usuario>

nombreLogin

2

Cambiar estado

3

Mostrar/Ocultar usuario

4

Dar de alta una película

Código de la película a dar de alta

<Poner aquí el resto de datos a pedir>

codPeliculaAlta

Dar de alta película

5

Dar de baja una película

Código de la película a dar de baja

<Visualizar aquí los datos de la peli>

codPeliculaBaja

6

Dar de baja esta película

7

Mostrar/Ocultar película

8

**Hacerlos con
DataList**

4

8

[Volver a la Página Principal](#)

FUNCIONES USUARIO REGISTRADO

Realice todas las funciones que quiera . Rellene los datos oportunos y pulse el botón.

Modificar mis datos personales

Nombre y apellidos

Dirección

nombre

Conchi Presedo

Modificar datos

1

Manuel Moreno Pitxitxi, 3

direccion

Aumentar mi crédito

Introduzca aquí la cantidad en euros

cantidadEuros

Aumentar crédito

0

Alquilar película

Seleccione de la lista la película a alquilar

9

<insertar aquí
DropDownList>

Alquilar

11

Devolver película

Seleccione de la lista la película a devolver

10

<insertar aquí
DropDownList con
los nombres de las
pelis>

Devolver

12

Hacer las
SELECT

Volver a la Página Principal

Transacción: Definición y contexto

- **Transacción:**
 - Unidad lógica de procesamiento de la BD que incluye una o más operaciones de acceso a la BD (inserción, eliminación, modificación o consulta).
 - Se limita con dos órdenes:
 - Comienzo: BEGIN TRANSACTION
 - Fin: END TRANSACTION
- **Contextos** en los que se utilizan transacciones:
 - **SGBDs multiusuario:** Muchos usuarios accediendo simultáneamente a la BD
 - Sistemas de procesamiento de transacciones
 - Sistemas con grandes BDs y cientos de usuarios concurrentes
 - Ej: sistemas de reserva, banca, cajas de supermercado...
 - En **procesos que consten de varios pasos** (que se deban dar todos) y/o haya accesos simultáneos a una base de datos. Ejemplos:
 - Alquilar una película.
 - Devolver una película.
 - Compras por Internet (en la siguiente página, pasos para comprar un billete de autobús).

Transacción: Definición y contexto (2)

Contratación de tu viaje

Paso: Datos personales y
asientos

1 Tu viaje 2 Elección de horario 3 > Datos personales y asientos 4 Resumen de la compra 5 Pago

DATOS DE LOS VIAJEROS

Nombre* Apellidos* N° Identidad* Tipo Documento

☐ JUAN LOPETEGUI 12345678H NIF

asiento de ida
BILBAO-BARAJAS T4

asiento de vuelta
BARAJAS T4-BILBAO

¡Recuerda identificarte como socio Busplus! Y si aún no eres Busplus, hazte socio y disfruta de todas sus ventajas

Importante:

una vez rellenados los datos proceda a elegir asiento marcando la opción de cada persona. Dispondrá de un tiempo de 20 minutos desde que escoge asiento hasta que finalice la transacción.

En el caso de que el viajero sea un menor indocumentado se debe indicar el documento de un adulto acompañante. [Condiciones de viaje de menores no acompañados](#)

DESCUENTOS APLICABLES

[Mas Información»](#)

☐ Familia Numerosa General
(20,0% de descuento)

☐ Familia Numerosa Especial
(50,0% de descuento)

Importante:

La selección de un descuento en este paso implica la aplicación de dicho descuento a todos los viajeros

ASIENTO DE IDA BILBAO-BARAJAS T4

01	02	04	03
05	06	08	07
09	10	12	11
13	14	16	15
17	18	20	19
21	22	24	23
25	26	30	29
27	28		
31	32		
35	36	34	33
39	40	38	37
43	44	42	41
47	48	46	45
51	52	50	49
55	56	54	53
59	60	58	57

ASIENTO DE VUELTA BARAJAS T4-BILBAO

03	02	01
06	05	04
09	08	07
12	11	10
15	14	13
18	17	
21	20	
24	23	22
27	26	25
30	29	28
33	32	31
36	35	34

asiento libre

asiento ocupado

asiento de su elección

Transacción: Propiedades ACID

- La utilización de transacciones acarrea **problemas de concurrencia** , que son aquéllos que se pueden producir cuando varios usuarios acceden a la vez a los mismos datos y alguno de ellos los puede modificar.
- Primer paso para minimizarlos: Propiedades ACID.

- **Propiedades ACID.**

Una transacción debe poseer estas propiedades:

- **Atomicidad** (Atomicity):

Garantiza que una transacción o se realiza por completo o no se realiza en absoluto.

- **Conservación de la consistencia** (Consistency):

Garantiza que una transacción lleva a la BD de un estado consistente a otro.

- **Aislamiento** (Isolability):

Garantiza que una transacción no interfiere con otras transacciones que se ejecuten concurrentemente.

- **Durabilidad** (Durability):

Garantiza que los cambios sobre la BD que realiza una transacción no se pierden aunque exista un fallo posterior.

Transacción: Propiedades ACID (2)

Atomicidad

- El conjunto de instrucciones SQL de la transacción se trata como una unidad lógica, es decir, o se ejecutan todas las instrucciones o no se ejecuta ninguna.
- Así, las instrucciones que contiene una transacción se hacen “temporalmente” hasta encontrar:
 - **COMMIT**: refleja todos los cambios en la base de datos (**confirma** las instrucciones)
 - **ROLLBACK**: no cambia nada de la base de datos (**deshace** todas las instrucciones)

Transacción: Propiedades ACID (3)

Conservación de la consistencia y Aislamiento

- Cada transacción ejecutada individualmente preserva la consistencia de la BD, pero una **ejecución concurrente** de varias transacciones no tiene por qué. El estándar SQL refiere tres posibles **problemas** de lectura cuando una transacción lee datos que podría haber cambiado otra:
 - Una **lectura sucia** ocurre cuando se permite a una transacción la lectura de una fila que ha sido modificada por otra transacción no confirmada todavía y que luego nunca se llega a confirmar.
 - Una **lectura no repetible** ocurre cuando en el curso de una transacción una fila se lee dos veces y los valores no coinciden, debido a que otra transacción la ha cambiado.
 - Una **lectura fantasma** ocurre cuando, durante una transacción, se ejecutan dos consultas idénticas y los resultados no coinciden. Es un caso particular de las lecturas no repetibles cuando la transacción repite una consulta acotada en rango *SELECT... WHERE* y, entre ambas operaciones otra transacción crea nuevas filas (en la misma tabla) que entran dentro de esa cláusula *WHERE*.

Transacción: Propiedades ACID (4)

Conservación de la consistencia y Aislamiento

- El aislamiento más estricto que garantiza siempre la consistencia sería ejecutar las operaciones de cada transacción de manera consecutiva, sin intercalar operaciones entre ellas → No hay concurrencia → Inaceptable en la práctica.
- Para conseguir consistencia y también concurrencia, el SGBD relacional generalmente relaja la propiedad ACID de aislamiento.
 - Para obtener el mayor nivel de aislamiento, utiliza diferentes **técnicas para el control de la concurrencia**:
 - generalmente hace un bloqueo de los datos de la BD a los que se accede (a distintos niveles: toda la tabla, filas concretas, rangos) o
 - implementa un control de concurrencia mediante versiones múltiples
 - Estas técnicas pueden resultar en una pérdida de concurrencia. Por ello se necesita añadir lógica adicional al programa que accede a los datos → Niveles de Aislamiento

Transacción: Propiedades ACID (5)

Conservación de la consistencia y Aislamiento

Los **niveles de aislamiento** están definidos en el estándar SQL y controlan el grado de bloqueo durante el acceso a los datos. De más bajo a más alto son:

- **Lecturas no confirmadas** (READ UNCOMMITTED) → Las instrucciones pueden leer filas que han sido modificadas por otras transacciones y que todavía no se han confirmado. No evita ninguno de los 3 problemas de lectura. Es el menor nivel de aislamiento.
- **Lecturas confirmadas** (READ COMMITTED) → Las instrucciones no pueden leer datos que hayan sido modificados pero aún no confirmados por otras transacciones. Evita las lecturas sucias, pero no las otras dos. Suele ser la opción predeterminada en muchos SGBDs.
- **Lecturas repetibles** (REPEATABLE READ) → Las instrucciones no pueden leer datos que han sido modificados pero aún no confirmados por otras transacciones y ninguna otra transacción puede modificar los datos leídos por la transacción actual hasta que ésta finalice. Evita las lecturas sucias y las no repetibles.
- **Serializable** (SERIALIZABLE) → Todas las transacciones ocurren de modo aislado, como si todas se ejecutaran de modo serie (una tras otra). Evita los tres tipos de problemas de lectura. Es el nivel de aislamiento más restrictivo.

Transacción: Código VB.NET para tecnología OLEDB (MS Access)

' PASO 1. CREAR Y ABRIR LA CONEXIÓN

```
Dim conexion As OleDbConnection  
conexion=New OleDb.OleDbConnection (cadenaconexion)  
conexion.Open()
```

...

' DECLARAR VARIABLES PARA TRANSACCION Y SUS INSTRUCCIONES EN SQL

```
Dim transaccion As OleDbTransaction
```

```
Dim strSQL As String
```

```
Dim dbComm As OleDbCommand
```

...

' CREAR Y COMENZAR LA TRANSACCION

```
transaccion = conexion.BeginTransaction()
```

DIFERENCIAS CON LA
CODIFICACIÓN DE
UNA SOLA
INSTRUCCIÓN SQL:
SOLO CAMBIA LO QUE
APARECE EN NEGRITA

Transacción: Código VB.NET (2)

Try

' CODIFICAR LA TRANSACCIÓN

...

```
' Preparar el comando con la instrucción SQL
strSQL = "XXXX"
dbComm = New OleDbCommand (strSQL, conexion, transaccion)
' Añadir parámetros si es necesario
dbComm.Parameters.AddXXX
' Ejecutar el comando SQL adecuado para la instrucción SQL
dbComm.ExecuteXXX()
```

...

' REFLEJAR LOS CAMBIOS EN LA BD SI TODO HA IDO BIEN

transaccion.Commit()

Catch ex As Exception

' Opcional para ver el mensaje de error

MsgBox (ex.Message)

' DESHACER TODO SI ALGO FUE MAL

transaccion.Rollback()

(PASOS 2 Y 3)
QUE SE DEBEN
REPETIR PARA CADA
INSTRUCCIÓN SQL
DE LA TRANSACCIÓN

End Try

Transacción: Código VB.NET (3)

...

' PASO 4. CERRAR LA CONEXION Y LIBERAR MEMORIA

conexion.Close()

conexion.Dispose()

El modo AUTOCOMMIT

- El modo autocommit significa que **toda instrucción SQL que se ejecute tiene su propia transacción implícita**, si la base de datos las soporta. Cuando este modo está activo, cada SQL (por ejemplo cada INSERT, UPDATE o DELETE) es una transacción, es decir, el resultado de la instrucción se refleja en la BD automáticamente nada más ser ejecutada y no se puede deshacer.
- **Para programar nuestras propias transacciones este modo debe estar desactivado.** En muchos casos, el método .BeginTransaction() de la conexión lo desactiva automáticamente.

Transacción: Ejemplo

- Instrucciones SQL que se necesitan para el botón **Alquilar** (en página 25) :
 - 1 SELECT para ver si la película es gratis o no.
 - Si no es gratis
 - 1 SELECT para comprobar que el saldo del usuario es suficiente si la película es de pago,
 - Si tiene saldo suficiente: un UPDATE para cobrarle el precio de la película (el recargo si se retrasa se le cobra al devolverla, no ahora. Se puede sacar un mensaje indicando al usuario que tiene 2 días para verla sin recargo.)
 - En cualquier caso (tanto gratis como pagando)
 - Un UPDATE para actualizar el estado de la película a 'Alquilada'
 - un INSERT para el nuevo alquiler. (si la película tiene límite de visionado, es decir, es de pago ya se verá cuando la vaya a devolver y se le cobrará el recargo si hace falta).

Algunas Webs con ejemplos

- Ejemplos de transacciones con Visual Studio
- [http://www.elguille.info/colabora/puntoNET/cone win Transacciones.htm](http://www.elguille.info/colabora/puntoNET/cone%20win%20Transacciones.htm)
- Ejemplos de código con OLEDB
- [http://www.vb-helper.com/howto net db transaction.html](http://www.vb-helper.com/howto_net_db_transaction.html)
- <https://msdn.microsoft.com/es-es/library/system.data.oledb.oledbconnection%28v=vs.71%29.aspx>
- <http://www.elguille.info/NET/library/System.Data.OleDb.aspx>
- Explicaciones de los métodos y la clase OLEDB
- <https://msdn.microsoft.com/en-us/library/system.data.oledb.oledbtransaction%28v=vs.110%29.aspx>