

Proyecto Final de Programación Web 1 - 2023

- [1. Tabla de calificación](#)
- [2. Base datos](#)
 - [2.1. SQL de la Propuesta 01](#)
 - [2.2. SQL de la Propuesta 02](#)
- [3. Descripción de tablas:](#)
- [4. Diccionario de datos \(Propuesta 1\)](#)
 - [4.1. DD de la tabla usuarios](#)
 - [4.2. DD de la tabla clientes](#)
 - [4.3. DD de la tabla tarjetas](#)
 - [4.4. DD de la tabla cuentas](#)
 - [4.5. DD para la tabla movimientos](#)
- [5. Simulación usando PHPMyAdmin](#)
 - [5.1. Inserción de usuarios](#)
 - [5.2. Inserción de clientes](#)
 - [5.3. Inserción de tarjetas](#)
 - [5.4. Apertura de cuentas](#)
 - [5.5. Realización de movimientos](#)
 - [5.6. Consulta del saldo](#)
- [6. Material](#)

Tema: Banca por internet:

1. Tabla de calificación

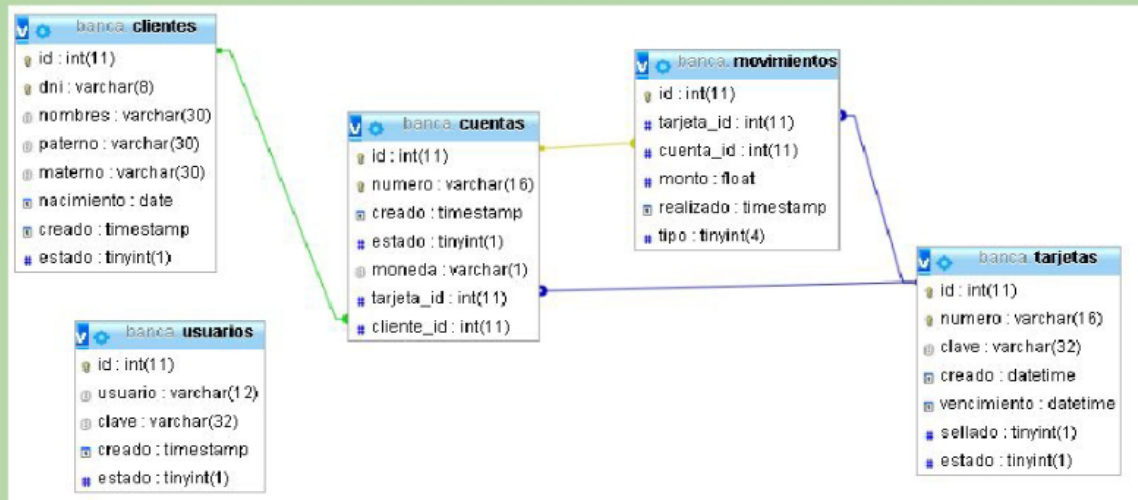
MySQL	<ul style="list-style-type: none">• Contruir una base de datos en MySQL para el proyecto.	3 pts
HTML + JS + CSS	<ul style="list-style-type: none">• Form1: Login para usuarios del banco (index.html)• Form2: Para registro de clientes (clientes.html)• Form3: Para registro de tarjetas (tarjetas.html)• Form4: Para registro de cuentas (cuentas.html)• Form5: Login para clientes (banca.html)• Form6: Para hacer Movimientos: depósitos, retiros. (movimientos.html)	7 pts
Perl + MySQL	<ul style="list-style-type: none">• HTML01: Muestra el estado: saldo y los últimos movimientos (estado.html)• Scripts para procesar los formularios: Form1, Form2, Form3, Form4, Form5 y Form6 (login.pl, clientes.pl, tarjetas.pl, cuentas.pl, banca.pl y movimientos.pl)• Script que muestra HTML01(estado.pl)	7 pts
Investigación	El alumno utiliza AJAX y/o Sesiones y lo aprovecha para el proyecto	3 pts
	TOTAL	20 pts

Fecha de presentación: Diciembre de 2023.

El proyecto representa el 18% de la nota final del curso (parte del 24% de EC3).

2. Base

datos **Propuesta 01:**



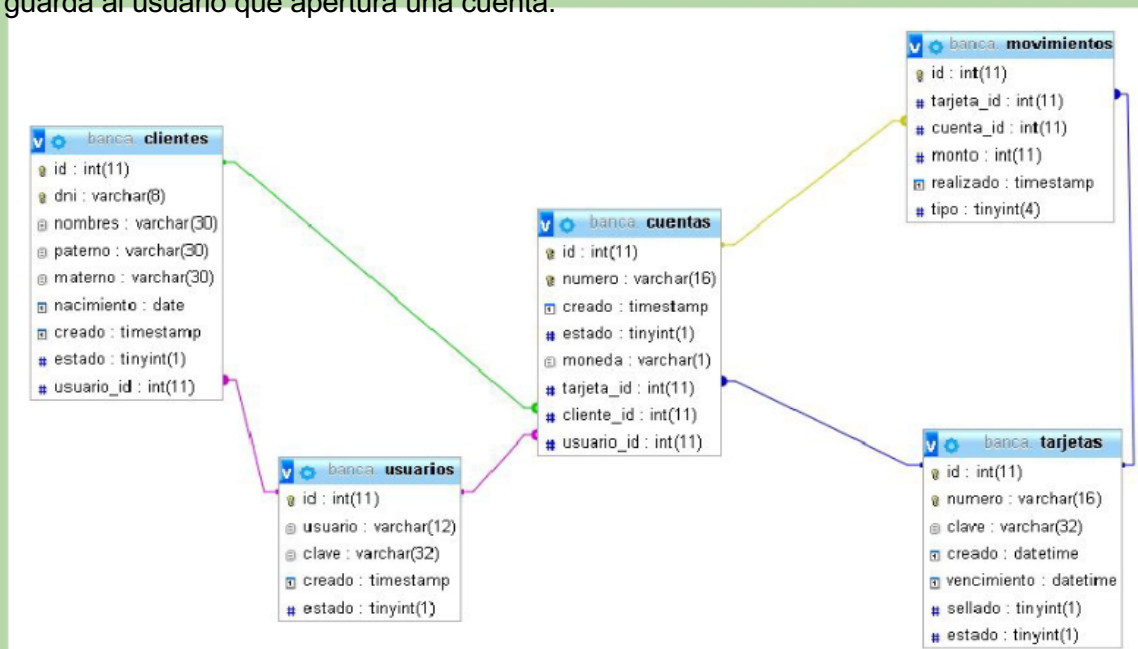
2.1. SQL de la Propuesta 01:

<https://drive.google.com/file/d/0B-oymxWlQkreRzdkZGxYdml4a1k/view?usp=sharing>
(Anexo A)

Propuesta 02: En esta propuesta de guarda se agregan las relaciones.

Se guarda al usuario que crea un nuevo cliente Se

guarda al usuario que apertura una cuenta.



2.2. SQL de la Propuesta 02:

<https://drive.google.com/file/d/0B-oymxWlQkreYjNDZHdBuK9tYVE/view?usp=sharing>
(Anexo B)

Por supuesto pueden haber muchas propuestas. Pero, con las que le hemos presentado puede empezar y despreocuparse por la BD. Por ejemplo los retiros los hace un cliente y los depósitos los hace un trabajador del banco o agente.

3. Descripción de tablas:

- **usuarios:** son los usuarios de la empresa bancaria, ellos con un usuario específico y una clave deberían poder realizar las siguientes tareas:
 - Ingresar nuevos clientes (ver tabla clientes)
 - Crear una nueva cuenta para un cliente asociada a una tarjeta (ver tablas cuentas y tarjetas). Recuerden que, primero existe el cliente y luego la cuenta.
 - Crear tarjetas nuevas (ver tabla tarjetas). Esta tabla hace de repositorio de tarjetas, no olvide que cuando apertura una cuenta en un banco, se obtiene una aleatoriamente y se le asocia a su cuenta.
- **clientes:** son las personas que tienen una cuenta en el banco, por lo tanto el banco les proporciona un número de cuenta y una tarjeta para poder realizar los movimientos en su cuenta. Los clientes usan el número de tarjeta que se les da y una clave que viene lacrada al momento de la apertura del sobre de la tarjeta.



- **cuentas:** son las cuentas bancarias para un determinado usuario. El número de cuenta es único en esta tabla, la moneda puede ser "s" para soles, "d" para dólares, etc. y el cliente_id es el cliente que le pertenece esta cuenta.
- **tarjetas:** Es sencillamente un repositorio de tarjetas. Por supuesto el número de tarjeta tiene 16 dígitos y es único en esta tabla, también una tarjeta viene con una clave que conoce el cliente.



- **movimientos:** Es el registro de los movimientos que un cliente hace en su cuenta, para ingresar a movimientos tuvo antes que haberse identificado con su tarjeta y clave de acceso.

Fecha

PARA MAYOR INFORMACION:
BANCA POR TELEFONO: (01)311-9898
BANCA POR INTERNET VIABCP WWW.VIABCP.COM

AGENTE BCP
YUDYFARMA

FECHA: 22/11/12 HORA: 20:43:40 H975373

NO. OPE: 313256

*Nro. de
Operación*

-----PAGO DE SERVICIOS-----
GIRO/RUBRO: EMPRESAS DIVERSAS
EMPRESA: INSTITUTO DE LA CONSTRUCCI
ON Y GERENCIA
CTA. A ABONAR: 1941142734066
COD. ID. USUARIO: 46441115
EN EFECTIVO

*Cod. de
Usuario*

DESCRIPCION
PAGOS VARIOS

IMPORTE PAGO:	S/.	130.00
CARGO FIJO:	S/.	0.00
MORA:	S/.	0.00

TOTAL DEUDA:	S/.	130.00
COMISION:	S/.	0.00
TOTAL A PAGAR:	S/.	130.00

Monto

4. Diccionario de datos (Propuesta 1)

4.1. DD de la tabla usuarios

usuarios

Columna	Tipo	Nulo	Predeterminado	Comentarios
id	int(11)	No		Código de usuario
usuario	varchar(12)	No		Nombre del usuario
clave	varchar(32)	No		Clave del usuario
creado	timestamp	No	CURRENT_TIMESTAMP	Fecha de creación del usuario
estado	tinyint(1)	No	1	Estado del usuario

Índices

Nombre de la clave	Tipo	Único	Empaquetado	Columna	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	id	2	A	No	

4.2. DD de la tabla clientes

clientes

Columna	Tipo	Nulo	Predeterminado	Comentarios
id	int(11)	No		Código del cliente
dni	varchar(8)	No		Nro de Dni del cliente
nombres	varchar(30)	No		Nombres del cliente
paterno	varchar(30)	No		Apellido paterno del cliente
materno	varchar(30)	No		Apellido materno del cliente
nacimiento	date	Sí	NULL	Fecha de nacimiento del cliente
creado	timestamp	No	CURRENT_TIMESTAMP	Fecha de creación del cliente
estado	tinyint(1)	No	1	Estado del cliente

Índices

Nombre de la clave	Tipo	Único	Empaquetado	Columna	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	id	0	A	No	
	BTREE	Sí	No	dni	0	A	No	

4.3. DD de la tabla tarjetas

tarjetas

Columna	Tipo	Nulo	Predeterminado	Comentarios
id	int(11)	No		Código de la tarjeta
numero	varchar(16)	No		Número de la tarjeta
clave	varchar(32)	No		Clave de la tarjeta
creado	datetime	No	CURRENT_TIMESTAMP	Fecha de creación de la tarjeta
vencimiento	datetime	Sí	NULL	Fecha de vencimiento de la tarjeta
sellado	tinyint(1)	No	1	Estado sellado de la tarjeta. Al inicio es 1. Si es 0 ya fue abierta.
estado	tinyint(1)	No	1	Estado de la tarjeta

Índices

Nombre de la clave	Tipo	Único	Empaquetado	Columna	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	id	0	A	No	
	BTREE	Sí	No	numero	0	A	No	

4.4. DD de la tabla cuentas

cuentas

Columna	Tipo	Nulo	Predeterminado	Enlaces a	Comentarios
id	int(11)	No			Código de la cuenta
numero	varchar(16)	No			Número de la cuenta
creado	timestamp	No	CURRENT_TIMESTAMP		Fecha de apertura de la cuenta
estado	tinyint(1)	No	1		Estado de la cuenta
moneda	varchar(1)	No	s		Tipo de moneda de la cuenta. 's' es soles, 'd' es dólares, etc.
tarjeta_id	int(11)	No		tarjetas -> id	Código de la tarjeta de esta cuenta
cliente_id	int(11)	No		clientes -> id	Código del dueño de esta cuenta

Índices

Nombre de la clave	Tipo	Único	Empaquetado	Columna	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	id	0	A	No	
numero	BTREE	Sí	No	numero	0	A	No	
cliente_id	BTREE	No	No	cliente_id	0	A	No	
tarjeta_id	BTREE	No	No	tarjeta_id	0	A	No	

4.5. DD para la tabla movimientos

movimientos

Columna	Tipo	Nulo	Predeterminado	Enlaces a	Comentarios
id	int(11)	No			Código del movimiento
tarjeta_id	int(11)	No		tarjetas -> id	Código de la tarjeta que hizo la operación
cuenta_id	int(11)	No		cuentas -> id	Código de la cuenta asociada a la operación
monto	float	No			Monto de la Operación
realizado	timestamp	No	CURRENT_TIMESTAMP		Fecha de la operación
tipo	tinyint(4)	No			Tipo de la operación. 1 es depósito, -1 es retiro.

Índices

Nombre de la clave	Tipo	Único	Empaquetado	Columna	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	id	0	A	No	
tarjeta_id	BTREE	No	No	tarjeta_id	0	A	No	
				cuenta_id	0	A	No	
cuenta_id	BTREE	No	No	cuenta_id	0	A	No	

5. Simulación usando PHPMysqlAdmin

La simulación nos ayuda a verificar si nuestra BD propuesta es funcional.

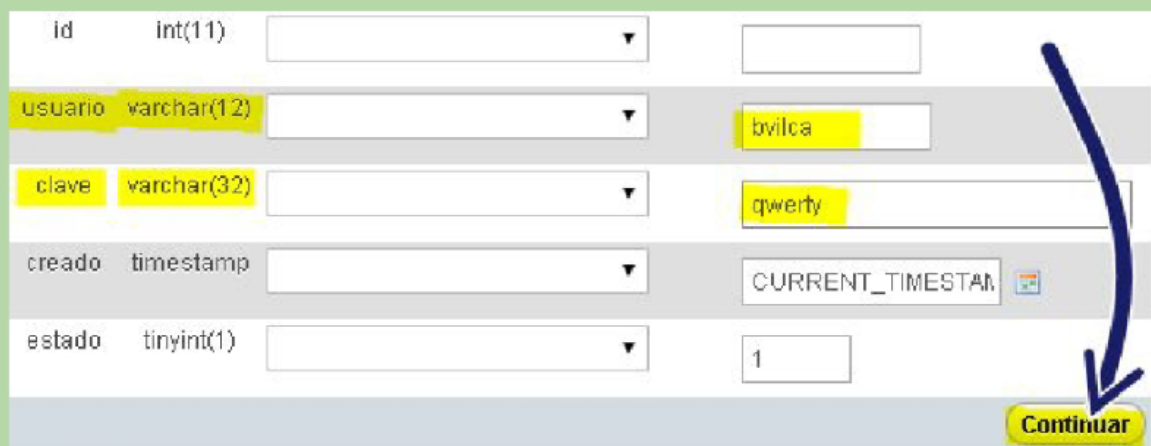
Todo se inicia cuando un determinado usuario (trabajador del banco) hace login al sistema.

5.1. Inserción de usuarios

Inserción de usuarios gráficamente

Para insertar un usuario primero sólo es necesario especificar el nombre de usuario y la clave.

Especificación de 2 atributos requeridos, los demás usan sus valores por defecto:



Claves encriptadas

La clave puede estar guardada sin encriptar por ahora, pero por temas de seguridad sería conveniente guardarlo encriptado. Ejemplo: MD5 genera 32 caracteres. Este es sólo una recomendación. Para presentar su prototipo del sistema banca no es necesario preocuparse ahora por esto.

Inserción de usuarios por código SQL

La consulta SQL para insertar un usuario es:

```
INSERT INTO usuarios (usuario, clave) VALUES ('rescobedo', 'qwerty');
```

Donde:

- **usuarios:** es el nombre de tabla donde se insertará.
- **usuario, clave:** los los atributos que definiremos en la consulta.
- **VALUES():** aquí se ingresan los valores que se aplicarán a los atributos especificados. Respetar el tipo de atributo. Por ejemplo un VARCHAR siempre esta encerrado en comillas simples.

Tanto la inserción gráfica o por ejecución de código SQL debieron generar en total 2 registros nuevos de usuarios:

consulta SQL: `SELECT * FROM `usuarios` LIMIT 0, 25 ;`
Filas: 2

id	usuario	clave	creado	estado
1	bwilca	qwerty	2014-12-05 10:19:04	1
2	rescobedo	qwerty	2014-12-05 10:19:16	1

Ahora tenemos 2 usuarios que pueden:

- [Registrar nuevos clientes]
- [Aperturar cuentas para los clientes]
- [Crear tarjetas]

5.2. Inserción de clientes:

Inserción de clientes gráficamente:

Para insertar un usuario sólo es necesario especificar el dni, los nombres, apellido paterno, apellido materno, el código del usuario que guarda este cliente (Ver propuesta 2).

Especificación de 5 atributos requeridos, los demás usan sus valores por defecto:

Columna	Tipo	Función	Nulo	Valor
id	int(11)	<input type="text"/>		<input type="text"/>
dni	varchar(8)	<input type="text"/>		70713456
nombres	varchar(30)	<input type="text"/>		Juan
paterno	varchar(30)	<input type="text"/>		Perez
materno	varchar(30)	<input type="text"/>		Tejada
nacimiento	date	<input type="text"/>	<input checked="" type="checkbox"/>	<input type="text"/>
creado	timestamp	<input type="text"/>		CURRENT_TIMESTAMP
estado	tinyint(1)	<input type="text"/>		1
usuario_id	int(11)	<input type="text"/>		1

 **Continuar**

Inserción de clientes por código SQL

```
INSERT INTO clientes (dni, nombres, paterno, materno, usuario_id)
VALUES ( '29551788', 'Rosa', 'Carpio', 'Mendoza', 1 );
```

Tanto la inserción gráfica o por ejecución de código SQL debieron generar en total 2 registros nuevos clientes:

consulta SQL: `SELECT * FROM `clientes` LIMIT 0, 25 ;`
Filas: 2


id	dni	nombres	paterno	materno	nacimiento	creado	estado	usuario_id
1	70713456	Juan	Perez	Tejada	NULL	2014-12-06 06:57:21	1	1 [->]
2	29551788	Rosa	Carpio	Mendoza	NULL	2014-12-06 07:04:50	1	1 [->]

5.3. Inserción de tarjetas

Inserción de tarjetas gráficamente:

Especificación de 2 atributos requeridos, los demás usan sus valores por defecto:

Columna	Tipo	Función	Nulo	Valor
id	int(11)	<input type="text"/>		<input type="text"/>
numero	varchar(16)	<input type="text"/>		4557880159472848
clave	varchar(32)	<input type="text"/>		123456
creado	datetime	<input type="text"/>		CURRENT_TIMESTAMP
vencimiento	datetime	<input type="text"/>	<input checked="" type="checkbox"/>	<input type="text"/>
sellado	tinyint(1)	<input type="text"/>		1
estado	tinyint(1)	<input type="text"/>		1

 **Continuar**

Observe que aquí hay un atributo "vencimiento" ya que toda tarjeta lo tiene. Sin embargo si no queremos considerar este requerimiento para una primer prototipo se le especifica como predeterminado tipo NULL.

Para aplicar una fecha de vencimiento por ejemplo 5 años de acuerdo a la fecha de creación (atributo "creado" en la tabla tarjetas se puede implementar un disparador conocido como TRIGGERS). Sin embargo no es necesario que nos preocupemos en este momento.

Como ya hemos mencionado estamos usando claves sin encriptar por ahora.

Inserción de tarjetas por código SQL

```
INSERT INTO tarjetas(numero, clave)
VALUES ( '4557880159472849' , '123456' ) ;
```

Ahora tenemos 2 tarjetas selladas para utilizarlas en la apertura de cuentas.

consulta SQL: SELECT * FROM 'tarjetas' LIMIT 0, 25 ;

Filas: 2

id	numero	clave	creado	vencimiento	sellado	estado
1	4557880159472848	123456	2014-12-06 07:14:41	NULL	1	1
2	4557880159472849	123456	2014-12-06 07:22:48	NULL	1	1

5.4. Apertura de cuentas

Apertura de cuentas a clientes (Gráficamente)

Especificación de 4 atributos requeridos, los demás usan sus valores por defecto:

Columna	Tipo	Función	Nulo	Valor
id	int(11)	<input type="text"/>		<input type="text"/>
numero	varchar(16)	<input type="text"/>		1941142734066
creado	timestamp	<input type="text"/>		CURRENT_TIMESTAMP
estado	tinyint(1)	<input type="text"/>		1
moneda	varchar(1)	<input type="text"/>		s
tarjeta_id	int(11)	<input type="text"/>		1
cliente_id	int(11)	<input type="text"/>		1
usuario_id	int(11)	<input type="text"/>		1



Continuar

Apertura de cuentas a clientes (Código SQL)

```
INSERT INTO cuentas (numero, tarjeta_id, cliente_id, usuario_id)
VALUES ( '1941142734067' , 2 , 2 , 2 ) ;
```

consulta SQL: SELECT * FROM 'cuentas' LIMIT 0, 25 ;

Filas: 2

id	numero	creado	estado	moneda	tarjeta_id	cliente_id	usuario_id
1	1941142734066	2014-12-06 07:29:19	1	s	1 [->]	1 [->]	1 [->]
2	1941142734067	2014-12-06 07:32:34	1	s	2 [->]	2 [->]	2 [->]

5.5. Realización de movimientos

Realización de movimientos (Gráficamente)

Ejemplo de un depósito de 200 soles

Columna	Tipo	Función	Nulo	Valor
id	int(11)	<input type="text"/>		<input type="text"/>
tarjeta_id	int(11)	<input type="text"/>	1	1
cuenta_id	int(11)	<input type="text"/>	1	1
monto	int(11)	<input type="text"/>		800
realizado	timestamp	<input type="text"/>		CURRENT_TIMESTAMP
tipo	tinyint(4)	<input type="text"/>	1	1

 **Continuar**

Realización de movimientos (Código SQL)

Ejemplo de un retiro de 200 soles

```
INSERT INTO movimientos (tarjeta_id, cuenta_id, monto, tipo)
VALUES (1, 1, 200, -1);
```

consulta SQL: SELECT * FROM `movimientos` LIMIT 0, 25 ;
Filas: 2

id	tarjeta_id	cuenta_id	monto	realizado	tipo
1	1 [->]	1 [->]	800	2014-12-06 07:38:38	1
2	1 [->]	1 [->]	200	2014-12-06 07:51:31	-1

5.6. Consulta del saldo

Para consultar el saldo se debe especificar la tarjeta y la cuenta a buscar. Esta consulta también es necesaria para evitar retiros superiores al saldo actual.

Vamos a averiguar el saldo del usuario Juan Perez cuya tarjeta es ID es "1" y cuenta ID es "1".

```
SELECT SUM(monto*tipo) FROM movimientos  
WHERE cuenta_id=1 AND tarjeta_id=1
```

La ejecución de esta consulta nos retornará el saldo:

SUM(monto*tipo)
600

Para el saldo es recomendable usar un ALIAS:

```
SELECT SUM(monto*tipo) AS 'saldo' FROM movimientos  
WHERE cuenta_id=1 AND tarjeta_id=1
```

La ejecución de esta consulta nos retornará el saldo:

saldo
600

Por lo que hemos podido ver la simulación nos dice que nuestra Base de datos es confiable.

Buena suerte en la elaboración del proyecto.

6. Material

Material para montar un servidor Web en maquina virtual GNU/Linux Ubuntu server:

-Use la máquina virtual proporcionada en la página Web del curso.

Material para montar un servidor Web en MS Windows:

<https://sites.google.com/site/hxampp/en-windows>

<https://sites.google.com/site/hxampp/perl-en-xampp>

<https://sites.google.com/site/hxampp/03-perl-dbi>

<https://sites.google.com/site/hxampp/04-perl-sessions>

<https://sites.google.com/site/hxampp/05-perl-ajax-mysql> etc.

Más material multimedia que puede interesar:

Tutorial CGI de Perl parte 5 - Crear sesiones con CGI::Session (Demo sistema login)

<https://www.youtube.com/watch?v=qtRRXy2oNUQ>

Tutorial CGI de Perl parte 4 - Conectar a una base de datos con DBI (Demo sistema de login)

<https://www.youtube.com/watch?v=HeyqidChWsw>

Tutorial CGI de Perl parte 3 - Enviando formularios (Métodos post y get)

<https://www.youtube.com/watch?v=GnuyEFickwo>

Tutorial CGI de Perl parte 2 - Usando métodos CGI para escribir código HTML

<https://www.youtube.com/watch?v=PBwl-H4GSBk>

Tutorial CGI de Perl parte 1 - Mi primer script CGI

<https://www.youtube.com/watch?v=xvclWylmSdo>