Juan David Torres A01702686

La caja de ahorro***s BDA_AD2020*** decide contratarlos para diseñar e implementar una base de datos distribuida para la administración de sus clientes, cuentas y empleados en cada una de las sucursales que tiene ya abiertas. La caja de ahorro empezó a funcionar hace 3 años (1 enero de 2017)

Las sucursales se ubican en las ciudades de Xalapa, Oaxaca, Estambul, Viena, Lisboa, Tokio, Cali y Santiago. Cada una de las sucursales tienen un código el cual se muestra en la siguiente tabla

Xalapa	XAL
Oaxaca	OAX
Estambul	EST
Viena	VNA
Lisboa	LIS
Tokio	TOK
Cali	COL
Santiago	STG

En cada una de las ciudades solamente se encuentra una sucursal en la cual se pueden abrir cuentas de ahorro y cuentas de tanda, cada depósito o retiro que se realiza se convierte a una moneda especial llamada DinarBDA (una aplicación convierte la moneda local a los DinarBDA y solo se registra el DinarBDA en la Base de datos)

- La cuenta de ahorro cobra una comisión de 0.9% por cada retiro, para los depósitos no tiene comisión.
  Mediante la aplicación

- La comisión es descontada del retiro, por ejemplo, si retiro 1000 DinarBDA me entregan 991 dinaBDA.
  Mediante la aplicación

- La cuenta tanda cobra una comisión de 0.59% por cada deposito, pero no cobra

por retiros.

- Entidad tipo de cuenta con comisión y movimiento con el monto

- La comisión se cobra del mismo deposito, por ejemplo, si deposito 1000 DinarDBA le descontamos la comisión y en la cuenta se ve reflejado un abono de 994.1.

· Una vez abierta la cuenta NUNCA se cierra y NUNCA puede tener saldo negativo, pero si 0

- Mediante la aplicación

· Si el cliente quiere realizar un retiro de su saldo y este no alcanza, se le entrega todo el saldo que tenga en ese momento y en dado caso se aplica su comisión. Quedando la cuenta en 0 pero activa para futuros depósitos.

- Mediante la aplicación

· En cualquiera de las cuentas si al 31 de diciembre de cada año tiene saldo a favor (mayor que 0) la caja realiza un depósito de un 5.123456789% del saldo promedio del año en curso. El depósito se ve reflejado con la leyenda SAF.

- Mediante la aplicación

· Si a lo largo del año no se retiró nada de lo depositado el 31 de diciembre del año en curso se realiza un depósito de un 9.123456789% del saldo promedio del año en curso, el depósito se ve reflejado con la leyenda SSR.

- Mediante la aplicación

· Los clientes pueden tener más de una cuenta y abrirla en más de una sucursal.

- Relación n a n

· Los empleados pueden trabajar solamente en una sucursal, sin embargo, pueden cambiar en algún momento del tiempo de sucursal.

- Agregando el idSucursal en la entidad de empleado

· Los clientes deciden de que cuenta retiran o depositan saldo sin importar si la cuenta pertenece (se abrió) a la sucursal.

- Movimiento entidad que relaciona cliente y cuenta

· En la sucursal de Xalapa se encuentran los directivos de la organización.

- Mediante la app

· Cada sucursal tiene un gerente y empleados que tienen un sueldo mensual.

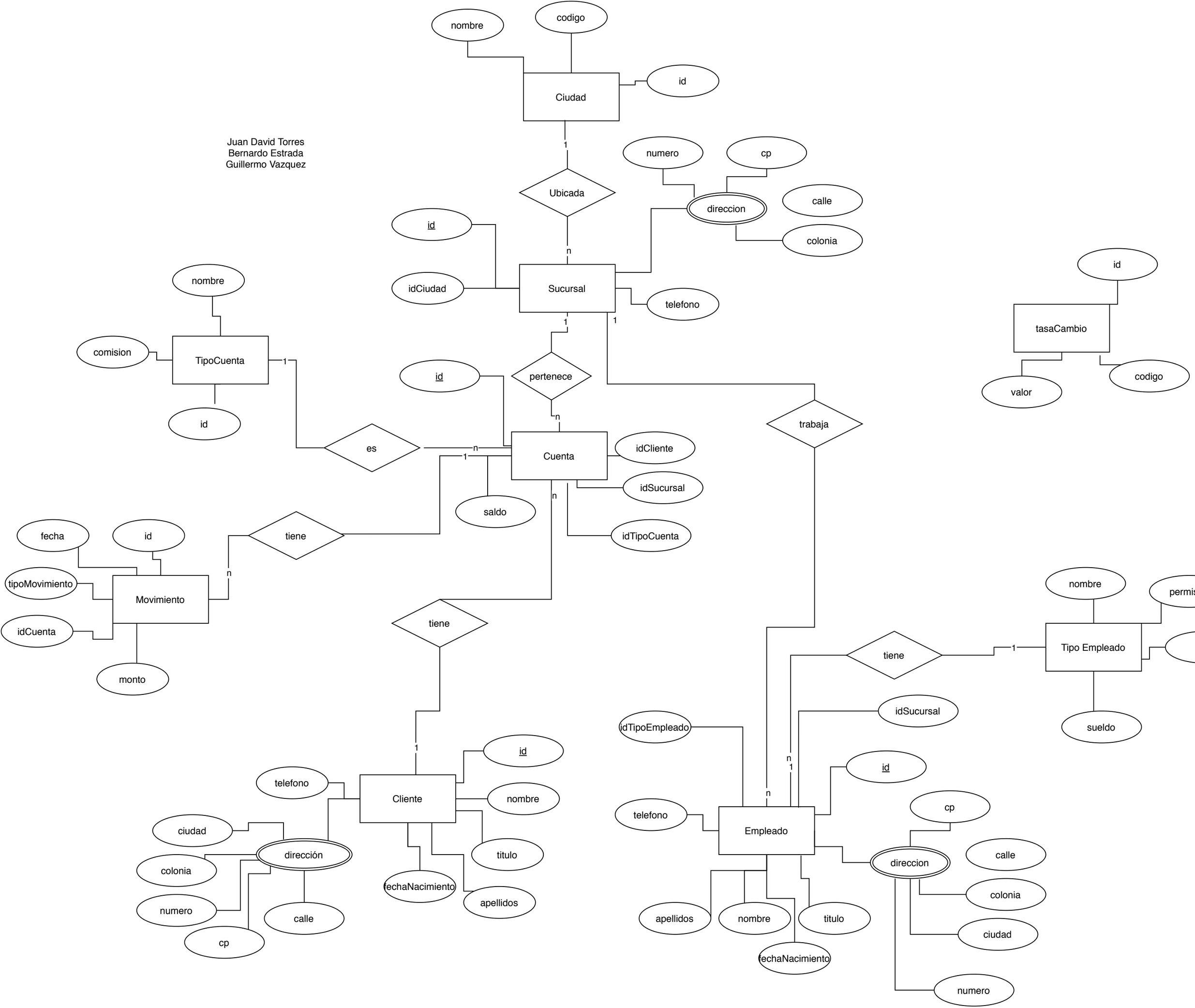
- Todos los gerentes tienen el mismo sueldo
- Todos los cajeros tienen el mismo sueldo
- Todos los gestores de cuenta tienen el mismo sueldo y son los únicos que pueden abrir cuentas.
    - Mediante la aplicación

· Las bases de datos guardan la información de dos sucursales y la localización de estas es:

- Xalapa – Xalapa y Oaxaca
- Estambul – Estambul y Viena
- Lisboa – Lisboa y Tokio
- Cali – Cali y Santiago

- Entidad de ciudad y fragmentación

Juan David Torres
Bernardo Estrada
Guillermo Vazquez



Juan David Torres
Bernardo Estrada
Guillermo Antonio Vazquez

Fragmentación

Esquema conceptual global

tasaCambio(id, valor, codigo)
cuenta(id, idCliente, idSucursal, idTipoCuenta, saldo)
tipoCuenta(id, comisión, nombre)
movimiento(id, fecha, tipoMovimiento, idCuenta, monto)
sucursal(id, telefono, direccion, idCiudad)
ciudad(nombre, codigo, id)
cliente(id, teléfono, direccion, fechaNacimiento, nombre, apellidos, titulo)
empleado(id, teléfono, nombre, apellidos, titulo, fechaNacimiento, direccion, idSucursal, idTipoEmpleado)
tipoEmpleado(id, nombre, permisos, sueldo)

Esquema de Fragmentación

Fragmento Cuentas

Relación Global

Cuenta (id, idCliente, idSucursal, idTipo Cuenta, saldo)

Fragmento Horizontal

Cuentas i: select * from Cuenta where Cuenta.idSucursal in (select S.idSucursal from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in (i))

Ejemplo

Cuentas 1: select * from Cuenta where Cuenta.idSucursal in (select S.id from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in (‘xal’, ‘oax’))

Cuentas 3: select * from Cuenta where Cuenta.idSucursal in (select S.id from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in (‘est’, ‘vna’))

Cuentas 5: select * from Cuenta where Cuenta.idSucursal in (select S.id from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in (‘lis’, ‘tok’))

Cuentas 7: select * from Cuenta where Cuenta.idSucursal in (select S.id from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in ('col', 'stg'))

Explicación

Esta fragmentación es para alojar la información de las cuentas que pertenecen a las sucursales que guarda el site. Esto es debido a que los sites guardan información de dos sucursales. Es un fragmento horizontal porque se obtiene toda la información y tiene una condicional.

Fragmento CuentasV

Fragmento Vertical

Nota: Es para obtener información necesaria sobre las cuentas a todos las sucursales

CuentasV i:

```
select id, cuenta, idTipoCuenta, saldo from Cuenta @j
Union
Select id, cuenta, idTipoCuenta, saldo from @k
Union
Select id, cuenta, idTipoCuenta, saldo from @l
```

Ejemplo

CuentasV 1 : select id, cuenta, idTipoCuenta, saldo from Cuenta @Estambul

```
Union
select id, cuenta, idTipoCuenta, saldo from @Lisboa
Union
select id, cuenta, idTipoCuenta, saldo from @Cali
```

CuentasV 3: select id, cuenta, idTipoCuenta, saldo from Cuenta @Xalapa

```
Union
select id, cuenta, idTipoCuenta, saldo from @Lisboa
Union
select id, cuenta, idTipoCuenta, saldo from @Cali
```

CuentasV 5: select id, cuenta, idTipoCuenta, saldo from Cuenta @Xalapa

```
Union
select id, cuenta, idTipoCuenta, saldo from @Estambul
Union
select id, cuenta, idTipoCuenta, saldo from @Cali
```

Cuentas V 7: select id, cuenta, idTipoCuenta, saldo from Cuenta @Xalapa

Union
select id, cuenta, idTipoCuenta, saldo from @Lisboa
Union
select id, cuenta, idTipoCuenta, saldo from @Estambul

Explicación

Esta fragmentación es para tener guardada la información necesaria de las cuentas en los sites que no pertenecen a la site donde fue creada y poder tener acceso más rápido al saldo de todas las cuentas por si quieren hacer retiros o depósitos desde otra sucursal. Es un fragmento vertical porque se elige que columnas desplegar.

Fragmento Tipo Cuenta

Relación Global

tipoCuenta(id, nombre, comision)

Fragmento Unico

TipoCuenta 1: select * from tipoCuenta

Ejemplo

SELECT * FROM tipoCuenta

Explicación

Es un fragmento único porque no va a cambiar y es necesario que se encuentre dentro de todos los sites porque todos los sites tienen cuentas que tienen tipos de cuentas.

Fragmento Movimiento

Relacion Global

Movimiento(id, fecha, tipoMovimiento, idCuenta, monto)

Fragmento Horizontal

Movimiento i: select * from Movimiento where Movimiento.idCuenta in
(select C.id from Cuenta C where idSucursal in
(select S.id from Sucursal S,Ciudad C
where S.idCiudad=C.id and C.codigo in (i))

Ejemplo

Nota: Ejemplo para alojar la información de ambas sucursales en el site Xalapa

Movimiento 1: select * from Movimiento where MovimientoidCuenta in
(select C.id from Cuenta C where idSucursal in
(select S.id from Sucursal S, Ciudad C

where S.idCiudad=C.id and C.codigo in (‘xal’, ‘oax’))

Explicación

Esta fragmentación se utiliza para alojar los movimientos presentes en cada una de las sucursales en el site que le corresponda y ya si se quiere acceder a la información desde otra sucursal es solo hacer un copiado de la información, ya que si se aloja también en otros se estaria ocupando mucho espacio de almacenamiento y ocupa más memoria. Por eso es un fragmento horizontal para alojar toda la información de las sucursales a la que corresponda la cuenta y tiene una condicional.

Fragmento Cliente

Relacion Global

cliente(id, telefono, direccion, fechaNacimiento, nombre, apellidos, titulo)

Fragmento Horizontal

Cliente i: select * from Cliente where Cliente.id in
(select C.idCliente from Cuenta C where idSucursal in
(select S.id from Sucursal S, Ciudad C
where S.idCiudad=C.id and C.codigo in (i))

Ejemplo

Nota: Ejemplo para el alojamiento de los clientes en la sucursal Xalapa y Oaxaca

Cliente 1: select * from Cliente where Cliente.id in
(select C.idCliente from Cuenta C where idSucursal in
(select S.id from Sucursal S, Ciudad C
where S.idCiudad=C.id and C.codigo in (‘xal’, ‘oax’))

Explicación

Esta fragmentación se utiliza para alojar la información de los clientes en los sites de las sucursales que correspondan y ya si se quiere acceder a la información desde otra sucursal es solo hacer un copiado de la información, ya que si se aloja también en otros se estaria ocupando mucho espacio de almacenamiento y ocupa más memoria. Es un fragmento horizontal porque es necesario guardar toda la información de los clientes en los sites que correspondan y tiene una condicional.

Fragmento Empleado

Relacion Global

empleado(id, telefono, nombre, apellidos, titulo, fechaNacimiento, direccion, idSucursal, idTipoEmpleado)

Fragmento Horizontal

Empleado i: select * from Empleado where Empleado.idSucursal in (select S.idSucursal from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in (i))

Ejemplo

Nota: Fragmentación para el alojamiento de la información de los empleados en Xalapa y Oaxaca en Xalapa

Empleado 1: select * from Empleado where Empleado.idSucursal in (select S.id from Sucursal S,Ciudad C where S.idCiudad=C.id and C.codigo in ('xal', 'oax'))

Explicación

Es un fragmento para alojar toda la información de los empleados de la sucursal a la que corresponda al site y ya si se quiere acceder a la información desde otra sucursal es solo hacer un copiado de la información, ya que si se aloja también en otros se estaría ocupando mucho espacio de almacenamiento y ocupa más memoria. Es un fragmento horizontal porque se obtiene toda la información de todas las columnas.

Fragmento Sucursal

Relacion Global

sucursal(id, telefono, direccion, idCiudad)

Fragmento Horizontal

Sucursal i:

select * from Sucursal S, Ciudad C where S.idCiudad = C.id and C.codigo in (i)

Ejemplo

Sucursal 1:

select * from Sucursal S, Ciudad C where S.idCiudad = C.id and C.codigo in ('xal','oax')

Explicación

Es un fragmento que se utiliza para alojar la información de las sucursales a las que corresponde el site y ya si se quiere acceder a la información desde otra sucursal es solo hacer un copiado de la información, ya que si se aloja también en otros se estaria ocupando mucho espacio de almacenamiento y ocupa más memoria. Es un fragmento horizontal porque se obtiene toda la información de la sucursal y tiene una condicional.

Fragmento Ciudad

Relacion Global

ciudad(nombre, codigo, id)

Fragmento Único

Ciudad 1: SELECT * from Ciudad

Ejemplo

Ciudad 1: SELECT * from Ciudad

Explicación

Esta fragmentación se utiliza para guardar todas las ciudades. Es un fragmento único porque no hay necesidad de separar la información en más de un sitio

Fragmento Tasa de Cambio

Relacion Global

tasaCambio(id, valor, codigo)

Fragmento Único

tasaDeCambio 1: select * from tasaCambio

Ejemplo

tasaDeCambio 1: select * from tasaCambio

Explicación

Es un fragmento único porque es un catálogo y es necesario para todos los sites almacenar la información.

Fragmento Tipo Empleado

Relacion Global

tipoEmpleado(id, nombre, permisos, sueldo)

Fragmento Unico

tipoEmpleado 1: select * from tipoEmpleado

Ejemplo

tipoEmpleado 1: select * from tipoEmpleado

Explicación

Es un fragmento único porque es información que se necesita en todos los sites, ya que todos los sites tienen empleados y estos tienen tipo de empleado. Además es una tabla que no cambia.

Esquema de alojamiento

Fragmentación / Site	Site 1 (xal-oax)	Site 3 (est-vna)	Site 5 (lis-tok)	Site 7 (col-stg)
Cuentas 1	*			
Cuentas 3		*		
Cuentas 5			*	
Cuentas 7				*
CuentasV 1	*			
CuentasV 3		*		
CuentasV 5			*	
CuentasV 7				*
TipoCuenta 1	*	*	*	*
Movimiento 1	*			
Movimiento 3		*		
Movimiento 5			*	
Movimiento 7				*
Cliente 1	*			
Cliente 3		*		
Cliente 5			*	
Cliente 7				*
Empleado 1	*			
Empleado 3		*		
Empleado 5			*	

Empleado 7				*
tasaDeCambio 1	*	*	*	*
Sucursal 1	*			
Sucursal 3		*		
Sucursal 5			*	
Sucursal 7				*
Ciudad 1	*	*	*	*
tipoEmpleado1	*	*	*	*

1. Site Xalapa-Oaxaca
3. Site Estambul-Viena
5. Site Lisboa-Tokio
7. Site Cali-Santiago

	Juan David Torres	**Bernardo Estrada**	**Guillermo Vazquez**
Autoevaluación	**100**	**100**	**100**
Coevaluación	**100**	**100**	**100**

Juan David Torres
A01702686

Reporte individual del aprendizaje, coevaluación y autoevaluación

Durante este semestre aprendí la importancia de las bases de datos en la vida de una aplicación y en la vida cotidiana. Aprendí los elementos de un modelo entidad relación y como con estos componentes se puede lograr un diseño muy detallado de una base de datos, esto ultimo lo aprendí en clase como también en el desarrollo del proyecto. Aprendí sobre bases de datos de datos distribuidas, la importancia de la fragmentación para no tener todo centralizado y no depender de otros sitios. Aprendí sobre los diferentes esquemas de bases de datos distribuidas como top-down y bottom up. Aprendí también sobre los diferentes tipos de lenguaje para hacer operaciones en SQL y sus funcionalidades. También sobre las propiedades ACID y como son de importantes para hacer transacciones, igual de importantes que el two phase commit para transacciones en bases de datos distribuidas.

Aprendí a sacar un MER partiendo de una descripción de un problema y modificar el diseño dependiendo de las reglas de negocio del cliente las cuales son de suma importancia para el buen funcionamiento. También entendí que es muy importante hacer solo lo que nos pide el cliente para que no haya malentendidos.

El desempeño de la clase podría mejorar utilizando los kahoots no como actividad sino como una forma de medir nuestros conocimientos y saber que se puede reforzar pero sin calificar. También enseñar un ejemplo de lo que deberíamos hacer en la practica con respecto al proyecto para que al momento de tener una actividad ya saber como debería ser y que elementos debería tener. Dar puntos extras por poner la cámara podría ser bueno, la forma en la que se da la clase es muy dinámica, si me despierta y entiendo muchos mas cosas que si solo se estuviera dando una lectura o algo por el estilo.

Yo como alumno podría mejorar mi rendimiento estudiando lo que vi el mismo día y también dormirme mas temprano para poder estar mas activo al momento de la clase. Siempre le presto atención al profesor en todo lo que dice y entiendo muy muy los conceptos entonces creo que mi rendimiento como estudiante ha sido muy bueno.

Siento que la clase esta muy interesante y que podría serlo aun mas utilizando tecnologías que se están utilizando hoy en día como bases de datos no relacionales, o tecnologías como graphql que son nuevas herramientas que están surgiendo y que se utilizan en nuevo software. También podría ser mas interesante si se mostraran ejemplos de la vida real de como se aplican los conocimientos que estamos adquiriendo en la clase.