Contenido:

- Integridad Relacional
- Operaciones de Álgebra Relacional
- Operaciones Tradicionales Teoría de conjuntos
- Ejercicios
- Creación de Tablas de datos
- Restricciones
- Insert, delete y update.
- Ejercicios

ÁLGEBRA RELACIONAL – INTEGRIDAD REFERENCIAL

➢ INTEGRIDAD REFERENCIAL

La integridad referencial es una propiedad deseable en las bases de datos. Gracias a la integridad referencial se garantiza que una entidad (fila o registro) siempre se relaciona con otras entidades válidas, es decir, que existen en la base de datos. Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas.

Todas las bases de datos relacionales gozan de esta propiedad gracias a que el software gestor de base de datos vela por su cumplimiento. En cambio, las bases de datos jerárquicas requieren que los programadores se aseguren de mantener tal propiedad en sus programas.

✓ FUNCIONAMIENTO

Supongamos una base de datos con las entidades Persona y Factura. Toda factura corresponde a una persona y solamente una. Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas.

Supongamos que una persona se identifica por su atributo DNI (Documento nacional de identidad). También tendrá otros atributos como el nombre y la dirección. La entidad Factura debe tener un atributo DNI_cliente que identifique a quién pertenece la factura.

Por sentido común es evidente que todo valor de DNI_cliente debe corresponder con algún valor existente del atributo DNI de la entidad Persona. Esta es la idea intuitiva de la integridad referencial.

Existen tres tipos de integridad referencial:

- Integridad referencial débil: si en una tupla de R todos los valores de los atributos de K tienen un valor que no es el nulo, entonces debe existir una tupla en S que tome esos mismos valores en los atributos de J.

- Integridad referencial parcial: si en una tupla de R algún atributo de K toma el valor nulo, entonces debe existir una tupla en S que tome en los atributos de J los mismos valores que los atributos de K con valor no nulo.

- Integridad referencial completa: en una tupla de R todos los atributos de K deben tener el valor nulo o bien todos tienen un valor que no es el nulo y entonces debe existir una tupla en S que tome en los atributos de J los mismos valores que toman los de K.

✓ PRIMARY KEY

La clave principal (PRIMARY KEY) nos permite asegurar la integridad de entidad (puesto que es única en cada registro) y por otro lado nos garantiza la estabilidad de las relaciones con otras tablas.

✓ FOREIGN KEY

La restricción FOREIGN KEY, se conoce como la clave externa o foránea que ya hemos explicado. Y como ya sabes es la pareja de la restricción PRIMARY KEY, y juntas cumplen con la integridad referencial.

Una clave externa es una copia de la clave principal de la tabla principal, se inserta en la tabla que se pretende enlazar y con esto creamos la relación entre un par de tablas. Las claves externas pueden ser varias en una misma tabla, mientras que las principales deben ser únicas.

Para que esta relación que comentamos se cumpla, la clave principal que enlaza con la externa debe cumplir obligatoriamente que las dos columnas sean del mismo tipo.

PRESENTACIÓN

Esta guía didáctica es un material de ayuda institucional, perteneciente a las especialidades de computación, ***Ingeniería de Software*** e ***Ingeniería de Redes y Comunicaciones*** tiene por finalidad proporcionar los conocimientos las técnicas de Modelamiento de Base de Datos a los estudiantes del segundo ciclo de estudios.

La **Organización SISE,** líder en la enseñanza tecnológica a nivel superior, promueve la elaboración de materiales educativos, en concordancia a las exigencias de las tecnologías de estos tiempos, que permiten la creación de nuevas herramientas de aprendizaje con el objetivo de facilitar el acceso de los estudiantes a la educación en el marco del desarrollo tecnológico de la informática u de las telecomunicaciones.

Esta guía se divide en 9 temas principales, las cuales se irán desarrollando por medio de contenidos especialmente preparados para un mejor aprendizaje del educando. Permite conocer las herramientas indispensables para la elaboración de diagramas de diseño de datos con el uso del Modelo Entidad relación. Se inicia con la descripción de conceptos básicos en las cuales se tiene por objetivo que el alumno se introduzca en lo concerniente a Base de datos.

En el proceso de desarrollo de sistemas informáticos, orientados a producir software que apoye a las actividades empresariales, así como a sus procesos, se tienen que respetar ciertas fases propias de las metodologías del análisis de información, de la metodología de procesos de negocios (IDEF) hoy en día se emplea la metodología orientada a objetos, sin embargo, para el desarrollo de software es primordial el manejo del análisis y diseño de sistemas, para el análisis tenemos herramientas de recopilación de información, mientras que para la fase de diseño de sistemas, tenemos las llamadas herramientas ‘CASE’, que son el apoyo informático de todo diseñador de sistemas para plasmar todo el análisis de requerimientos previos en diagramas, conocidos como ‘MODELOS’, la herramientas de diseño de sistemas más empleado es el Platinum Erwin, que es la que emplearemos para el curso.

Este material en su primera edición, servirá para ayudar a nuestros estudiantes SISESINOS a tener una formación solida para resolver casos y problemáticas presentados en una organización empresarial.

✓ FUNCIONAMIENTO

Supongamos una base de datos con las entidades Persona y Factura. Toda factura corresponde a una persona y solamente una. Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas.

Supongamos que una persona se identifica por su atributo DNI (Documento nacional de identidad). También tendrá otros atributos como el nombre y la dirección. La entidad Factura debe tener un atributo DNI_cliente que identifique a quién pertenece la factura.

Por sentido común es evidente que todo valor de DNI_cliente debe corresponder con algún valor existente del atributo DNI de la entidad Persona. Esta es la idea intuitiva de la integridad referencial.

Existen tres tipos de integridad referencial:

- Integridad referencial débil: si en una tupla de R todos los valores de los atributos de K tienen un valor que no es el nulo, entonces debe existir una tupla en S que tome esos mismos valores en los atributos de J.

- Integridad referencial parcial: si en una tupla de R algún atributo de K toma el valor nulo, entonces debe existir una tupla en S que tome en los atributos de J los mismos valores que los atributos de K con valor no nulo.

- Integridad referencial completa: en una tupla de R todos los atributos de K deben tener el valor nulo o bien todos tienen un valor que no es el nulo y entonces debe existir una tupla en S que tome en los atributos de J los mismos valores que toman los de K.

✓ PRIMARY KEY

La clave principal (PRIMARY KEY) nos permite asegurar la integridad de entidad (puesto que es única en cada registro) y por otro lado nos garantiza la estabilidad de las relaciones con otras tablas.

✓ FOREIGN KEY

La restricción FOREIGN KEY, se conoce como la clave externa o foránea que ya hemos explicado. Y como ya sabes es la pareja de la restricción PRIMARY KEY, y juntas cumplen con la integridad referencial.

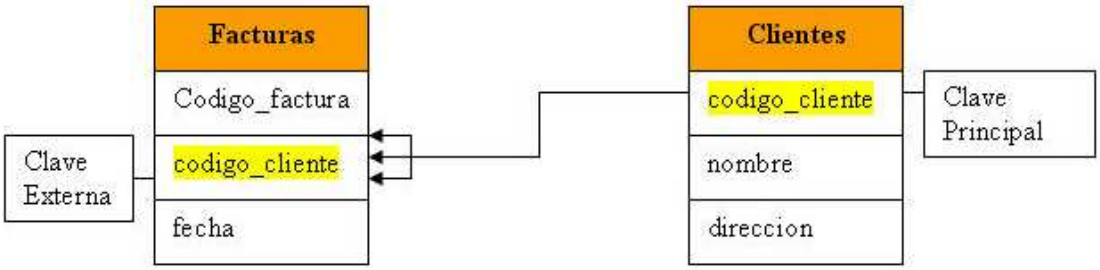
Una clave externa es una copia de la clave principal de la tabla principal, se inserta en la tabla que se pretende enlazar y con esto creamos la relación entre un par de tablas. Las claves externas pueden ser varias en una misma tabla, mientras que las principales deben ser únicas.

Para que esta relación que comentamos se cumpla, la clave principal que enlaza con la externa debe cumplir obligatoriamente que las dos columnas sean del mismo tipo.

✓ INTEGRIDAD REFERENCIAL EN CASCADA

Esta tipo de integridad que surgió con la versión 2000 de SQL Server, permite una serie de operaciones, que sólo pueden llevarse a cabo de este modo y no de otro.

Expliquemos porque a este tipo de integridad referencial se le añade el concepto de cascada. Imagina que tenemos una tabla que almacena las facturas emitidas para los clientes. Esta tabla entre sus campos contiene el campo codigo_cliente, que es la clave externa que se relaciona con la clave principal de la tabla clientes. Por lo tanto la tabla clientes tendrá un campo codigo_cliente que se relaciona con la tabla Facturas. Puedes verlo algo más claro en el siguiente gráfico:



Como ves, tenemos una relación uno a varios. Mientras en la tabla Clientes el campo codigo_cliente será único en cada registro, en la tabla Facturas, este mismo campo puede aparecer varias veces en diferentes registros, ya que emitimos varias facturas para el mismo cliente.

Esta relación representa una integridad referencial estricta, la cual no nos permite modificar el valor del código de cliente en la tabla clientes ya que de algún modo dejaría "huérfanos" a aquellos registros que anteriormente estaban relacionados al código de cliente que tratamos de modificar. Otra limitación que tenemos con esta integridad es que no nos permiten eliminar un cliente que tenga facturas emitidas en la tabla Facturas, por lo tanto no podemos eliminar un registro que tenga otros registros referenciados mediante estas relaciones.

Precisamente para solucionar estos problemas que hemos planteado tenemos la integridad referencial en cascada. Podemos añadir una serie de acciones que permita solventar estas complicaciones.

Podemos incluir actualizaciones en cascada, de modo que cuando modifiquemos el valor de una clave principal en la tabla principal, se modifiquen del mismo modo los valores de las claves externas en el resto de tablas enlazadas a la principal. En nuestro caso, al modificar el valor del campo codigo_cliente de un determinado cliente, este nuevo valor se cambiará para todos los registros referenciados en la tabla Facturas.

Igualmente podemos incluir eliminaciones en cascada, de tal forma que si eliminamos un registro de la tabla principal, se eliminen también los registros enlazados en la tabla subordinada. En nuestro caso, podríamos eliminar un cliente, y automáticamente se eliminarían todos sus registros de facturas.

Debes tener mucho cuidado a la hora de utilizar este tipo de relaciones en cascada, ya que si activamos por ejemplo la eliminación en cascada, corremos peligro de que un usuario no sea consciente de que perderá todos los registros vinculados a esa tabla.

Una clave principal, por norma general no cambiará su valor, por lo tanto no tiene sentido activar en esos casos tampoco la actualización en cascada.

✓ INTEGRIDAD REFERENCIAL EN CASCADA

Las restricciones de integridad referencial en cascada permiten definir las acciones que SQL Server 2005 lleva a cabo cuando un usuario intenta eliminar o actualizar una clave a la que apuntan las claves externas existentes.

Las cláusulas REFERENCES de las instrucciones CREATE TABLE y ALTER TABLE admiten las cláusulas ON DELETE y ON UPDATE. Las acciones en cascada también se puede definir mediante el cuadro de diálogo Relaciones de clave externa.

[ON DELETE { NO ACTION | CASCADE | SET NULL | SET DEFAULT }]

[ON UPDATE { NO ACTION | CASCADE | SET NULL | SET DEFAULT }]

✓ NO ACTION

Es el valor predeterminado si no se especifica ON DELETE u ON UPDATE.

✓ ON DELETE NO ACTION

Especifica que si se intenta eliminar una fila con una clave a la que hacen referencia las claves externas de las filas existentes en otras tablas, se produce un error y se revierte la instrucción DELETE.

✓ ON UPDATE NO ACTION

Especifica que si se intenta actualizar un valor de clave en una fila a cuya clave hacen referencia las claves externas de filas existentes en otras tablas, se produce un error y se revierte la instrucción UPDATE.

✓ CASCADE, SET NULL y SET DEFAULT

Permiten la eliminación o actualización de valores de clave de modo que se pueda realizar un seguimiento de las tablas definidas para tener relaciones de clave externa en la tabla en la que se realizan las modificaciones. Si las acciones referenciales en cascada se han definido también en las tablas de destino, las acciones en cascada especificadas se aplican para las filas eliminadas o actualizadas. No se puede especificar CASCADE para ninguna de las claves externas o principales que tengan una columna timestamp.

✓ ON DELETE CASCADE

Especifica que si se intenta eliminar una fila con una clave a la que hacen referencia claves externas de filas existentes en otras tablas, todas las filas que contienen dichas claves externas también se eliminan.

- ✓ **ON UPDATE CASCADE**

Especifica que si se intenta actualizar un valor de clave de una fila a cuyo valor de clave hacen referencia claves externas de filas existentes en otras tablas, también se actualizan todos los valores que conforman la clave externa al nuevo valor especificado para la clave.

Nota:

CASCADE no se puede especificar si una columna timestamp es parte de una clave externa o de la clave a la que se hace referencia.

- ✓ **ON DELETE SET NULL**

Especifica que si se intenta eliminar una fila con una clave a la que hacen referencia las claves externas de las filas existentes de otras tablas, todos los valores que conforman la clave externa de las filas a las que se hace referencia se establecen en NULL. Todas las columnas de clave externa de la tabla de destino deben aceptar valores NULL para que esta restricción se ejecute.

- ✓ **ON UPDATE SET NULL**

Especifica que si se intenta actualizar una fila con una clave a la que hacen referencia las claves externas de las filas existentes de otras tablas, todos los valores que conforman la clave externa de las filas a las que se hace referencia se establecen en NULL. Todas las columnas de clave externa de la tabla de destino deben aceptar valores NULL para que esta restricción se ejecute.

- ✓ **ON DELETE SET DEFAULT**

Especifica que si se intenta eliminar una fila con una clave a la que hacen referencia las claves externas de las filas existentes de otras tablas, todos los valores que conforman la clave externa de las filas a las que se hace referencia se establecen como predeterminados. Todas las columnas de clave externa de la tabla de destino deben tener una definición predeterminada para que esta restricción se ejecute. Si una columna acepta valores NULL y no se ha establecido ningún valor predeterminado explícito, NULL se convierte en el valor predeterminado implícito de la columna. Todos los valores distintos de NULL que se establecen debido a ON DELETE SET DEFAULT deben tener unos valores correspondientes en la tabla principal para mantener la validez de la restricción de la clave externa.

- ✓ **ON UPDATE SET DEFAULT**

Especifica que si se intenta actualizar una fila con una clave a la que hacen referencia las claves externas de las filas existentes de otras tablas, todos los valores que conforman la clave externa de la fila a los que se hace referencia se establecen en sus valores predeterminados. Todas las columnas externas de la tabla de destino deben tener una definición predeterminada para que esta restricción se ejecute. Si una columna se convierte en NULL, y no hay establecido ningún valor predeterminado explícito, NULL deviene el valor predeterminado implícito de la columna. Todos los valores no NULL que se establecen debido a ON UPDATE

los sistemas de base de datos incluyen por lo menos una de estas aplicaciones integradas.

- La mayoría de los sistemas proporcionan además interfaces integradas adicionales en las que los usuarios no emiten en absoluto solicitudes explícitas a la base de datos, sino que en vez de ello operan mediante la selección de elementos en un menú o llenando casillas de un formulario. Estas interfaces controladas por menús o por formularios tienden a facilitar el uso a personas que no cuentan con una capacitación formal en tecnología de la información (IT). En contraste, las interfaces controladas por comandos tienden a requerir cierta experiencia profesional en IT, aunque tal vez no demasiada. Por otra parte, es probable que una interfaz controlada por comandos sea más flexible que una controlada por menús o por formularios, dado que los lenguajes de consulta por lo regular incluyen ciertas características que no manejan esas otras interfaces.

- El administrador de base de datos o DBA.

Algunos usuarios son:

- Jefes de proyecto.
- Analistas de sistemas.
- Analistas programadores.
- Programadores.
- Diseñadores de sistemas.

✓ ADMINISTRADOR DE BASE DE DATOS

Es el profesional informático responsable de diseñar la estructura de la base de datos, así como del mantenimiento y seguridad tanto de la información como del servidor de datos. Su denominación es DBA (Database Administrator), entre sus funciones principales tenemos:

- Definición de la estructura de tablas, y componentes.
- Asignación y administración de permisos de acceso a los usuarios.
- Responsable de la seguridad de toda la información, por medio de copias de seguridad de datos (backups).
- Administrar la estructura de la Base de Datos
- Administrar la actividad de los datos
- Administrar el Sistema Manejador de Base de Datos
- Establecer el Diccionario de Datos
- Asegurar la confiabilidad de la Base de Datos
- Confirmar la seguridad de la Base de Datos
- Asegurar una óptima performance de la organización de los datos.

Detallemos algunos de ellos:

Administración de la estructura de la Base de Datos

La administración de la estructura de la Base de Datos incluye participar en el diseño inicial de la misma y su puesta en práctica así como controlar, y administrar sus requerimientos,

SET DEFAULT deben tener unos valores correspondientes en la tabla principal para mantener la validez de la restricción de clave externa.

Ejemplos.

CODCLIE	NOMBRE
C0001	RAUL LOPEZ
C0002	ABRAHAM JARA

NRO_FACTURA	FECHA_EMISION	CODCLIE	NOMBRE
F00011	20/12/2009	C0001	RAUL LOPEZ
F00012	16/01/2010	C0002	ABRAHAM JARA

En este caso tenemos dos tablas, en las cuales los clientes registrados en el sistema tienen facturas ya generadas producto de una compra que ellos realizaron. Pongámonos en los dos posibles casos de integridad referencial en cascada:

- Si queremos modificar el nombre del cliente en la tabla Cliente, entonces este dato debería actualizarse también en la tabla Factura, esto ocurrirá siempre y cuando hayamos añadido esta instrucción en el momento de la creación de la tabla. Quedaría así:

CODCLIE	NOMBRE
C0001	JUAN LAVADO
C0002	ABRAHAM JARA

NRO_FACTURA	FECHA_EMISION	CODCLIE	NOMBRE
F00011	20/12/2009	C0001	JUAN LAVADO
F00012	16/01/2010	C0002	ABRAHAM JARA

- Si queremos eliminar a un cliente en la tabla Cliente, entonces este dato debería supuestamente elimnarse también en la tabla Factura, esto ocurrirá siempre y cuando hayamos añadido esta instrucción en el momento de la creación de la tabla. ¿Pero, es correcto hacer esto?

CODCLIE	NOMBRE
C0002	ABRAHAM JARA

NRO_FACTURA	FECHA_EMISION	CODCLIE	NOMBRE
F00012	16/01/2010	C0002	ABRAHAM JARA

¡No es correcto!

No deberíamos eliminar facturas ya que esto puede generar conflictos de información más adelante, existe la llamada eliminación lógica, en la cual podemos crear un campo en la tabla

Cliente que indique si el cliente está activo o no (Podemos utilizar el tipo de dato bit, que maneja como valores el 0 y 1, es un tipo de dato boolean).

➢ ALGEBRA RELACIONAL

El álgebra relacional es un conjunto de operaciones que describen paso a paso como computar una respuesta sobre las relaciones, tal y como éstas son definidas en el modelo relacional. Denominada de tipo procedimental, a diferencia del Cálculo relacional que es de tipo declarativo.

Describe el aspecto de la manipulación de datos. Estas operaciones se usan como una representación intermedia de una consulta a una base de datos y, debido a sus propiedades algebraicas, sirven para obtener una versión más optimizada y eficiente de dicha consulta.

✓ OPERACIONES DEL ALGEBRA RELACIONAL

Las operaciones de álgebra relacional manipulan relaciones. Esto significa que estas operaciones usan uno o dos relaciones existentes para crear una nueva relación. Esta nueva relación puede entonces usarse como entrada para una nueva operación. Este poderoso concepto - la creación de una nueva relación a partir de relaciones existentes hace considerablemente más fácil la solución de las consultas, debido a que se puede experimentar con soluciones parciales hasta encontrar la proposición con la que se trabajará.

El álgebra relacional consta de nueve operaciones:

- Unión
- Intersección
- Diferencia
- Producto
- Selección
- Proyección
- Reunión
- División
- Asignación

Las cuatro primeras se toman de la teoría de conjunto de las matemáticas; las cuatro siguientes son operaciones propias del álgebra relacional y la última es la operación estándar de dar un valor a un elemento.

UNIÓN

La operación de unión permite combinar datos de varias relaciones. Supongamos que una determinada empresa internacional posee una tabla de empleados para cada uno de los países en los que opera. Para conseguir un listado completo de todos los empleados de la empresa tenemos que realizar una unión de todas las tablas de empleados de todos los países.

No siempre es posible realizar consultas de unión entre varias tablas, para poder realizar esta operación es necesario e imprescindible que las tablas a unir tengan las mismas estructuras, que sus campos sean iguales.

INTERSECCIÓN

La operación de intersección permite identificar filas que son comunes en dos relaciones. Supongamos que tenemos una tabla de empleados y otra tabla con los asistentes que han realizado un curso de inglés (los asistentes pueden ser empleados o gente de la calle). Queremos crear una figura virtual en la tabla denominada "Empleados que hablan Inglés", esta figura podemos crearla realizando una intersección de empleados y curso de inglés, los elementos que existan en ambas tablas serán aquellos empleados que han asistido al curso.

DIFERENCIA

La operación diferencia permite identificar filas que están en una relación y no en otra. Tomando como referencia el caso anterior, deberíamos aplicar una diferencia entre la tabla empleados y la tabla asistentes al curso para saber aquellos asistentes externos a la organización que han asistido al curso.

PRODUCTO

La operación producto consiste en la realización de un producto cartesiano entre dos tablas dando como resultado todas las posibles combinaciones entre los registros de la primera y los registros de la segunda. Esta operación se entiende mejor con el siguiente ejemplo:

Tabla A

X	Y
10	22
11	25

Tabla B

W	Z
33	54
37	98
42	100

El producto de A * B daría como resultado la siguiente tabla:

Tabla A * Tabla B			
10	22	33	54
10	22	37	98
10	22	42	100
11	25	33	54
11	25	37	98
11	25	42	100

SELECCIÓN

La operación selección consiste en recuperar un conjunto de registros de una tabla o de una relación indicando las condiciones que deben cumplir los registros recuperados, de tal forma que los registros devueltos por la selección han de satisfacer todas las condiciones que se hayan establecido. Esta operación es la que normalmente se conoce como consulta.

Podemos emplearla para saber que empleados son mayores de 45 años, o cuales viven en Madrid, incluso podemos averiguar los que son mayores de 45 años y residen en Madrid, los que son mayores de 45 años y no viven en Madrid, etc..

En este tipo de consulta se emplean los diferentes operadores de comparación (=,>, <, >=, <=, <>), los operadores lógicos (and, or, xor) o la negación lógica (not).

PROYECCIÓN

Una proyección es un caso concreto de la operación selección, esta última devuelve todos los campos de aquellos registros que cumplen la condición que he establecido. Una proyección es una selección en la que seleccionamos aquellos campos que deseamos recuperar. Tomando como referencia el caso de la operación selección es posible que lo único que nos interese recuperar sea el número de la seguridad social, omitiendo así los campos teléfono, dirección, etc.. Este último caso, en el que seleccionamos los campos que deseamos, es una proyección.

REUNIÓN

La reunión se utiliza para recuperar datos a través de varias tablas conectadas unas con otras mediante cláusulas JOIN, en cualquiera de sus tres variantes INNER, LEFT, RIGHT. La operación reunión se puede combinar con las operaciones selección y proyección.

Un ejemplo de reunión es conseguir los pedidos que nos han realizado los clientes nacionales cuyo importe supere 15.000 unidades de producto, generando un informe con el nombre del cliente y el código del pedido. En este caso se da por supuesto que la tabla clientes es diferente a la tabla pedidos y que hay que conectar ambas mediante, en este caso, un INNER JOIN.

DIVISIÓN

La operación división es la contraria a la operación producto y quizás sea la más compleja de explicar, por tanto comenzaré con directamente con un ejemplo. Una determinada empresa posee una tabla de comerciales, otra tabla de productos y otra con las ventas de los comerciales. Queremos averiguar que comerciales han vendido todo tipo de producto.

Lo primero que hacemos es extraer en una tabla todos los códigos de todos los productos, a esta tabla la denominamos A.

Tabla A

Código Producto
1035
2241
2249
5818

En una segunda tabla extraemos, de la tabla de ventas, el código del producto y el comercial que lo ha vendido, lo hacemos con una proyección y evitamos traer valores duplicados. El resultado podría ser el siguiente:

Tabla B

Código Comercial	Código Producto
10	2241
23	2518
23	1035
39	2518
37	2518
10	2249
23	2249
23	2241

Si dividimos la tabla B entre la tabla A obtendremos como resultado una tercera tabla que:

Los campos que contiene son aquellos de la tabla B que no existen en la tabla A. En este caso el campo Código Comercial es el único de la tabla B que no existe en la tabla A.

Un registro se encuentra en la tabla resultado si y sólo si está asociado en tabla B con cada fila de la tabla A

Tabla Resultado

Código Comercial
23

¿Por qué el resultado es 23?. El comercial 23 es el único de la tabla B que tiene asociados todos los posibles códigos de producto de la tabla A.

ASIGNACIÓN

Esta operación algebraica consiste en asignar un valor a uno o varios campos de una tabla.

✓ LAS OPERACIONES

Básicas

Cada operador del álgebra acepta una o dos relaciones y retorna una relación como resultado. σ y Π son operadores unarios, el resto de los operadores son binarios. Las operaciones básicas del álgebra relacional son:

SELECCIÓN (Σ)

Permite seleccionar un subconjunto de tuplas de una relación (**R**), todas aquellas que cumplan la(s) condición(es) **P**, esto es:

$$\sigma_P(R)$$

Ejemplo:

$$\sigma_{Apellido=Gomez}(Alumnos)$$

Selecciona todas las tuplas que contengan Gómez como apellido en la relación Alumnos.

Una condición puede ser una combinación booleana, donde se pueden usar operadores como: $\wedge, \vee$, combinándolos con operadores $<, >, \leq, \geq, =, \neq$.

PROYECCIÓN (Π)

Permite extraer columnas(atributos) de una relación, dando como resultado un *subconjunto vertical* de atributos de la relación, esto es:

$$\Pi_{A_1,A_2,\ldots,A_n}(R)$$

donde $A_1, A_2, \ldots, A_n$son atributos de la relación **R** .

Ejemplo:

$$\Pi_{Apellido,Semestre,NumeroControl}(Alumnos)$$

Selecciona los atributos Apellido, Semestre y NumeroControl de la relación Alumnos, mostrados como un subconjunto de la relación Alumnos

PRODUCTO CARTESIANO (X)

El producto cartesiano de *dos relaciones* se escribe como:

$$R \times S$$

y entrega una relación, cuyo *esquema* corresponde a una combinación de todas las tuplas de **R** con cada una de las tuplas de **S**, y sus atributos corresponden a los de **R** seguidos por los de **S**.

Ejemplo:

$$Alumnos \times Maestros$$

Muestra una nueva relación, cuyo esquema contiene cada una de las tuplas de la relación Alumnos junto con las tuplas de la relación Maestros, mostrando primero los atributos de la relación Alumnos seguidos por las tuplas de la relación Maestros.

UNIÓN (∪)

La operación

$$R \cup S$$

Retorna el conjunto de tuplas que están en R, o en S, o en ambas. R y S deben ser *uniones compatibles*.

DIFERENCIA (-)

La diferencia de dos relaciones, R y S denotada por:

$$R - S$$

Entrega todas aquellas tuplas que están en R, pero **no** en S. R y S deben ser *uniones compatibles*.

Estas operaciones son fundamentales en el sentido en que (1) todas las demás operaciones pueden ser expresadas como una combinación de éstas y (2) ninguna de estas operaciones pueden ser omitidas sin que con ello se pierda información.

No básicas

Entre los operadores no básicos tenemos:

INTERSECCIÓN (∩)

La intersección de dos relaciones se puede especificar en función de otros operadores básicos:

$$R \cap S = R - (R - S)$$

La intersección, como en Teoría de conjuntos, corresponde al conjunto de todas las tuplas que están en R y en S, siendo R y S *uniones compatibles*.

REUNIÓN NATURAL (⋈) (NATURAL JOIN)

La operación Reunión natural en el álgebra relacional es la que permite reconstruir las tablas originales previas al proceso de normalización. Consiste en combinar las proyección, selección

y producto cartesiano en una sola operación, donde la condición θ es la igualdad Clave Primaria = Clave Externa (o Foranea), y la proyección elimina la columna duplicada (clave externa).

Expresada en las operaciones básicas, queda

$$R \bowtie S = \Pi_{A1,A2...An}(\sigma_\theta(R \times S))$$

Una reunión zeta (θ-Join) de dos relaciones es equivalente a:

$$R \bowtie_\theta S = \sigma_\theta(R \times S)$$

Donde la condición θ es libre.

Si la condición θ es una igualdad se denomina EquiJoin.

DIVISIÓN (/)

Supongamos que tenemos dos relaciones A(*x, y*) y B(*y*) donde el dominio de *y* en A y B, es el mismo.

El operador división **A / B** retorna todos los distintos valores de *x* tales que para todo valor *y* en B existe una tupla $\langle x, y \rangle$ en A.

Ejemplos

Suponga las relaciones o tablas:

Alumno

ID	NOMBRE	CIUDAD	EDAD
01	Pedro	Santiago	14
11	Juan	Buenos Aires	18
21	Diego	Lima	12
31	Rosita	Concepción	15
41	Manuel	Lima	17

Apoderado

ID	NOMBRE	FONO	ID_ALUMNO
054	Víctor	654644	21
457	José	454654	11
354	María	997455	31
444	Paz	747423	01

Curso

COD	NOMBRE	FECHA_INICIO	DURACION	VALOR
01142	Sicología	13-01	15	3.000
02145	Biología	15-02	12	2.500
03547	Matemáticas	01-03	30	4.000
04578	Música	05-04	10	1.500
05478	Física	20-04	15	3.200

Inscrito

ID	ID_AL	COD
1	01	05478
2	01	02145
3	11	03547
4	21	02145
5	41	03547

Mostrar los nombres de los alumnos y su apoderado

Primero, realizaremos una **combinación** entre alumnos y apoderados (pues necesitamos saber a que alumno le corresponde tal apoderado). La combinación realizará un producto cartesiano, es decir, para cada tupla de alumnos (todas las filas de *alumnos*) hará una mezcla con cada una tupla de apoderados y seleccionará aquellas nuevas tuplas en que alumnos.id sea igual a apoderados.id_alumno, esto es:

ID (alumno)	NOMBRE (alumno)	CIUDAD	EDAD	ID (apoderado)	NOMBRE (apoderado)	FONO	ID_ALUMNO
01	Pedro	Santiago	14	054	Víctor	654644	21
01	Pedro	Santiago	14	457	José	454654	11
01	Pedro	Santiago	14	354	María	997455	31
01	*Pedro*	*Santiago*	*14*	*444*	*Paz*	*747423*	**01**
11	Juan	Buenos Aires	18	054	Víctor	654644	21
11	*Juan*	*Buenos Aires*	*18*	*457*	*José*	*454654*	**11**
11	Juan	Buenos Aires	18	354	María	997455	31
11	Juan	Buenos Aires	18	444	Paz	747423	01
21	*Diego*	*Lima*	*12*	*054*	*Víctor*	*654644*	**21**
21	Diego	Lima	12	457	José	454654	11
21	Diego	Lima	12	354	María	997455	31
21	Diego	Lima	12	444	Paz	747423	01
31	Rosita	Concepción	15	054	Víctor	654644	21
31	Rosita	Concepción	15	457	José	454654	11
31	*Rosita*	*Concepción*	*15*	*354*	*María*	*997455*	**31**
31	Rosita	Concepción	15	444	Paz	747423	01
41	Manuel	Lima	17	054	Víctor	654644	21
41	Manuel	Lima	17	457	José	454654	11
41	Manuel	Lima	17	354	María	997455	31
41	Manuel	Lima	17	444	Paz	747423	01

Por tanto, el resultado final de la combinación es:

$$\text{Alumnos} \bowtie_{Alumnos.ID = Apoderados.ID\_ALUMNO} \text{Apoderados}$$

ID (alumno)	NOMBRE (alumno)	CIUDAD	EDAD	ID (apoderado)	NOMBRE (apoderado)	FONO	ID_ALUMNO
01	Pedro	Santiago	14	444	Paz	747423	01
11	Juan	Buenos Aires	18	457	José	454654	11
21	Diego	Lima	12	054	Víctor	654644	21
31	Rosita	Concepción	15	354	María	997455	31

Ahora, aquí debemos mostrar solo el nombre del alumno y el nombre del apoderado, esto lo hacemos con un **Proyect** *o Proyección*, donde la tabla final sería:

$$\Pi_{Alumnos.NOMBRE, Apoderados.NOMBRE}$$

NOMBRE (alumno)	NOMBRE (apoderado)
Pedro	Paz
Juan	José
Diego	Víctor
Rosita	María

Resumiendo en un solo paso:

$\Pi_{Alumnos.NOMBRE, Apoderados.NOMBRE}($ **Alumnos** $\triangleleft\triangleright_{Alumnos.ID = Apoderados.ID\_ALUMNO}$ **Apoderados**$)$

Se lee: Proyecta los nombre de alumnos y nombre de apoderados de los alumnos cuyo ID sea el mismo que el ID_ALUMNO de los apoderados.

Mostrar el nombre de los alumnos inscritos y el nombre de los cursos que tomaron *[editar]*

Comenzaremos con una combinación entre los inscritos y los cursos para obtener el nombre de los cursos:

$$Inscritos \bowtie_{Inscritos.COD = Cursos.COD} Cursos$$

Lo que nos da la tabla:

Resultado 1

ID	ID_AL	COD (inscritos)	COD (cursos)	NOMBRE	FECHA_INICIO	DURACION	VALOR
1	01	05478	05478	Física	20-04	15	3.200
2	01	02145	02145	Biología	15-02	12	2.500
3	11	03547	03547	Matemáticas	01-03	30	4.000
4	21	02145	02145	Biología	15-02	12	2.500
5	41	03547	03547	Matemáticas	01-03	30	4.000

Como podemos observar, la **combinación** solo nos entrega las combinaciones entre *Inscritos* y *Cursos* en que *COD* sea igual entre los inscritos y el curso correspondiente.

Ahora necesitamos los nombres de los alumnos inscritos. Al resultado anterior (*Resultado 1*) aplicaremos una nueva **combinación** comparando los *ID* de los alumnos para colocar el nombre adecuado con el estudiante adecuado:

Resultado 1 $\bowtie_{Resultado\ 1.ID\_AL = Alumnos.ID}$ **Alumnos**

O escrito todo junto:

$($**Inscritos** $\bowtie_{Inscritos.COD = Cursos.COD}$ **Cursos**$)$ $\bowtie_{Resultado\ 1.ID\_AL = Alumnos.ID}$ **Alumnos**

La tabla de este nuevo resultado sería:

Resultado 2

ID (inscrito)	ID_AL	COD (inscritos)	COD (cursos)	NOMBRE (curso)	FECHA_INICIO	DURACION	VALOR	ID (alumno)	NOMBRE (alumno)	CIUDAD	EDAD
1	01	05478	05478	Física	20-04	15	3.200	01	Pedro	Santiago	14
2	01	02145	02145	Biología	15-02	12	2.500	01	Pedro	Santiago	14
3	11	03547	03547	Matemáticas	01-03	30	4.000	11	Juan	Buenos Aires	18
4	21	02145	02145	Biología	15-02	12	2.500	21	Diego	Lima	12
5	41	03547	03547	Matemáticas	01-03	30	4.000	41	Manuel	Lima	17

Finalmente con una **Proyección** mostraremos el nombre del alumno y el curso inscrito:

$$\Pi_{Resultado2.NOMBRE\ (alumno),Resultado2.NOMBRE\ (curso)}(\textbf{Resultado 2})$$

Donde la tabla final sería:

NOMBRE (alumno)	NOMBRE (curso)
Pedro	Física
Pedro	Biología
Juan	Matemáticas
Diego	Biología
Manuel	Matemáticas

La expresión completa sería:

$$\Pi_{Resultado2.NOMBRE\ (alumno),Resultado2.NOMBRE\ (curso)}((\textbf{Inscritos}\ ' _{Inscritos.COD = Cursos.COD} Cursos)\ ' _{Resultado1.ID\_AL = Alumnos.ID}\ \textbf{Alumnos})$$

Mostrar los nombres y precios de los cursos inscritos con valor menor a 3.000 *[editar]*

$$\Pi_{Resultado1.NOMBRE,\ Resultado1.VALOR}(\delta_{Cursos.VALOR<3000}(\textbf{Cursos}))$$

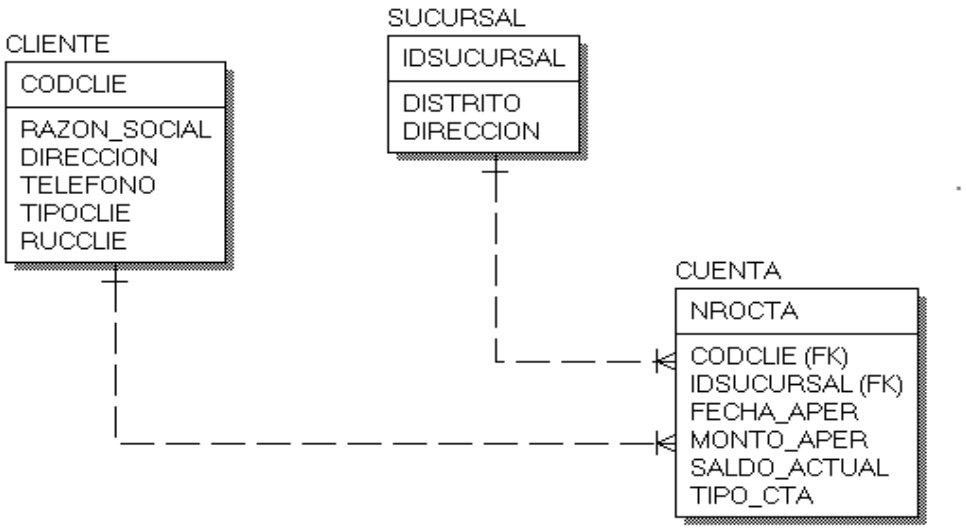
Lo que nos entregaría la tabla:

Resultado final

NOMBRE	VALOR
Biología	2.500
Música	1.500

➢ EJERCICIOS DE CREACIÓN DE TABLAS

Crear las siguientes tablas:



Aplicar las siguientes reglas de integridad de datos:

- Todo cliente debe tener un número de ruc irrepetible.

```
create table cliente(
codclie char(5),
razon_social varchar(50),
direccion varchar(50),
telefono char(7),
tipoclie varchar(30),
rucclie char(11) UNIQUE
CONSTRAINT PK_CLIE PRIMARY KEY(codclie)
)
```

- Agregar las claves foráneas a la tabla Cuentas. El tipo de cuenta puede tener sólo los valores I,II y III. Además debe tener a la fecha actual del sistema como valor por defecto para el campo fecha_aper.

```
create table sucursal(
idsucursal char(8) primary key,
distrito varchar(20),
```

```
direccion varchar(40)
)

create table cuentas(
nrocta char(8) primary key,
codclie char(5) constraint fk_cta1 foreign key(codclie) references cliente(codclie),
idsucursal char(8),
fecha_aper datetime default getdate(),
monto_aper decimal(8,2),
saldo_actual decimal(8,2),
tipo_cta varchar(20) CONSTRAINT CHK_CTA CHECK(TIPO_CTA IN('I','II','III'))
CONSTRAINT FK_CTA1 FOREIGN KEY(CODCLIE) REFERENCES CLIENTE(CODCLIE),
CONSTRAINT FK_CTA2 FOREIGN KEY(IDSUCURSAL) REFERENCES SUCURSAL(IDSUCURSAL)
)
```

EJERCICIOS PARA RESOLVER

Interprete las siguientes reglas de negocio en tablas con restricciones.

- Las cuotas de pago para un crédito pueden ser de 3, 6, 12 y 24 meses.
- Para registrar a un nuevo socio del club, la cuota de inscripción debe ser mayor o igual a 500 nuevos soles.
- Los únicos turnos disponibles para matriculas de alumnos son mañana y tarde.

LABORATORIO # 7

Crear los modelos en Erwin y la posterior base de datos en SQL Server de los siguientes casos de estudio:

CASO 1

La empresa de formación X, desea llevar un control informatizado de los cursos que imparte así como de lo profesores que participan en dichos cursos. Para ello, nos han dado las siguientes especificaciones:

• Cada curso, del que se desea conocer el título, el número de horas y el tema o los temas que trata, se identifica por un código de cuso.

• Cada curso puede tener una serie de cursos cuyo realización previa es obligatoria (prerrequisito) o recomendada.

• Cada curso se puede impartir una o varias veces, en diferentes fechas y en cada edición del mismo pueden participar diferentes empleados.

• Los empleados, de los que se desea conocer su código de empleado, nombre, DNI y fecha de antigüedad en la empresa, pueden impartir y recibir cursos pero con la restricción de que en una mismo edición de un curso no pueden participar como profesores y como alumnos.

CASO 2

La empresa Personal Quality desea incorporar en su política de contratación criterios de calidad del personal basados en la medición de sus habilidades o competencias.

• La empresa desea medir las competencias intelectuales de todos sus empleados y además desea conocer las competencias emocionales de sus directivos (por ejemplo, la capacidad de trabajo en grupo, la motivación, capacidad de liderazgo, etc.). De todas ellas se desea conocer: su código de identificación, su nombre y su descripción. Además, para cada competencia emocional se desea conocer, lo que se ha denominado el umbral; es decir, el valor mínimo de cada competencia por debajo del cual ningún empleado podrá ser directivo. Se requiere también que todo directivo mantenga este umbral mínimo en, al menos, 5 competencias emocionales.

• Para llevar a cabo este estudio, Personal Quality ha contactado con el Emotional Skill Center quien le ha proporcionado una batería de Test. Cada competencia está asociada a un conjunto de test que permiten medirla. Un test puede medir una única competencia. Cada test se identifica por un nombre y debe tener asociado un conjunto de preguntas, una plantilla para su corrección así como el modo en que se deberán interpretar los resultados.

• Cada empleado se identifica por un código interno. Además se quiere conocer el nombre, la dirección y un teléfono de contacto de cada empleado.

CASO 3

La gestión de una farmacia requiere poder llevar control de los medicamentos existentes, así como de los que se van sirviendo, para lo cual se pretende diseñar un sistema acorde a las siguientes especificaciones:

• En la farmacia se requiere una catalogación de todos los medicamentos existentes, para lo cual se almacenará un código de medicamento, nombre del medicamento, tipo de medicamento (jarabe, comprimido, pomada, etc.), unidades en stock, unidades vendidas y precio. Existen medicamentos de venta libre, y otros que sólo pueden dispensarse con receta médica.

• La farmacia adquiere cada medicamento a un laboratorio, o bien los fabrica ella misma. Se desea conocer el código del laboratorio, nombre, teléfono, dirección, fax así como el nombre de la persona de contacto.

• Los medicamentos se agrupan en familias, dependiendo del tipo de enfermedades a las que dicho medicamento se aplica.

• La farmacia tiene algunos clientes que realizan los pagos de sus pedidos a fin de cada mes (clientes con crédito). La farmacia quiere conocer las unidades de cada medicamento comprado (con o sin crédito) así como la fecha de compra. Además, es necesario tener los datos bancarios de los clientes con crédito, así como la fecha de pago de las compras que realizan.

Semana 8