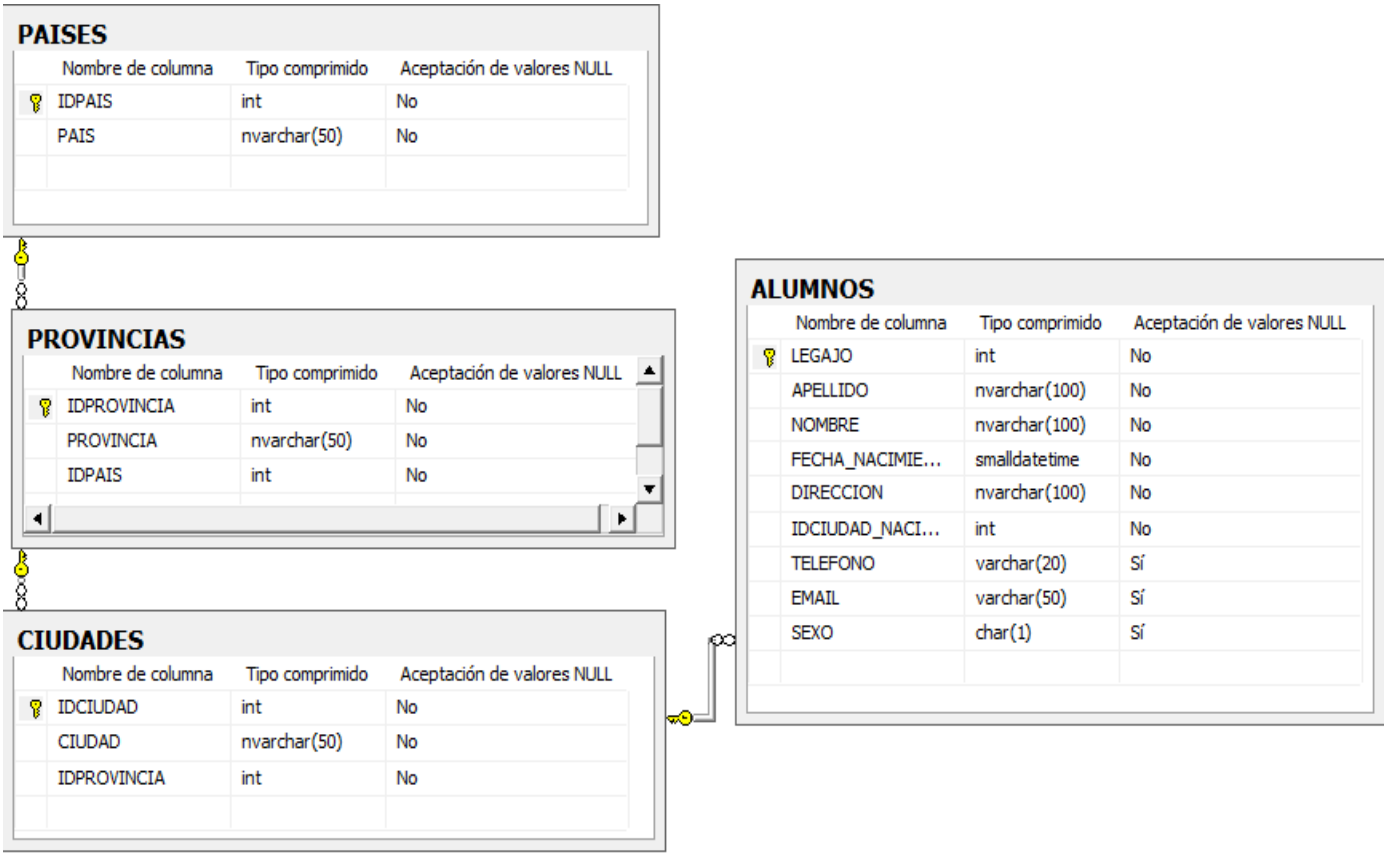
Carrera: Técnico Superior en Programación
Materia: Laboratorio de Computación III
Tema: Consultas de acción

Consultas de acción

Las consultas de acción permiten cambiar el estado de los datos en las tablas existentes en la base de datos. Las mismas pertenecen al tipo DML ya que sólo modifican los datos y no la estructura de los mismos.
Para los ejemplos que figuran en el siguiente apunte, utilizaremos la siguiente base de datos:



Insertar datos

La inserción de datos en una base de datos mediante código SQL se realiza mediante la palabra reservada INSERT.
La inserción se ejecutará con éxito si no existe ninguna incompatibilidad con los datos y no existe una restricción del tipo check o del tipo primary o foreign key.

La forma general para una consulta del tipo INSERT es la siguiente:
`INSERT INTO <tabla> [campo1, campo2, campoN, ... ] VALUES [valor1, valor2, valorN, ...]`

Ejemplos:

Insertando un país a la tabla países

```
INSERT INTO paises (pais) VALUES ('Colombia')
```

En este caso se ignora el campo idPais ya que el mismo es autoincrementable por lo que no hay que incorporarlo a la consulta de insert. Este tipo de consulta de insert, indica qué campos se les escribirán datos.

Veamos un ejemplo de inserción en la tabla ciudades

```
INSERT INTO ciudades (ciudad, idprovincia) VALUES ('Olivos', 1)
```

En este caso se ingresan los valores para dos campos. El primero es el nombre de la ciudad por lo que mientras el valor no sea nulo no habrá problemas. El segundo es el idprovincia, el cual exige integridad referencial con el campo idProvincia en la tabla de provincias. Por lo que se deberá ingresar un valor que exista en dicha tabla.

También se puede insertar múltiples filas en la misma consulta de INSERT (Funciona en SQL Server 2008+)

```
INSERT INTO ciudades (ciudad, idprovincia) VALUES ('Martinez', 1), ('Escobar',
1), ('Garín', 1), ('Benavidez', 1)
```

Inserción de un registro en la tabla de alumnos

```
INSERT INTO alumnos (legajo, apellido, nombre, fecha_nacimiento, direccion,
idciudad_nacimiento, telefono, sexo)
VALUES (9000, 'Fernandez', 'Juan', '1987/05/08', 'Uruguay 1234', 1, NULL, 'M')
```

En este ejemplo, ingresamos los valores para todos los campos de la tabla excepto para el campo *email*. El mismo como acepta nulo no provocará ningún problema al realizar la inserción.
Para el resto de los campos se ingresa su respectivo tipo de dato en la forma que corresponde con la aclaración de que el campo legajo debe no repetirse y el campo idciudad_nacimiento debe contener un valor existente en la tabla ciudades.

Modificar datos

La modificación de datos se realiza a través de la consulta de UPDATE. Hay que prestar especial atención en que cuando uno realiza una modificación debe indicar sobre qué registro/s se deberá realizar. Esto quiere decir que si no le indicamos algún tipo de restricción la modificación de

datos se realizará en todos los registros!
Para discriminar el conjunto de registros a modificar se utiliza la cláusula WHERE tal y como se utilizaba en las consultas de SELECT para excluir un conjunto de registros de la lista que se deseaba obtener.
Todos los registros que cumplan la/s condición/es de la cláusula WHERE serán modificados.

La forma general de una consulta del tipo UPDATE es la siguiente:
UPDATE <tabla> SET campo1 = valor [, campo2=valor, campoN=valor] WHERE [condición]

Ejemplos:

Modificar el email a NULL a todos los alumnos que hayan nacido entre 1980 y 1985

```
UPDATE alumnos SET email = NULL WHERE YEAR(fecha_nacimiento) BETWEEN 1980 AND 1985
```

Modificar el nombre a 'Juan Carlos' y la dirección a 'Belgrano 4567' al alumno con legajo 9000

```
UPDATE alumnos SET nombre = 'Juan Carlos', direccion = 'Belgrano 4567' WHERE legajo = 9000
```

Modificar el teléfono a NULL a todos los registros

```
UPDATE alumnos SET telefono = NULL
```

Como se puede apreciar en los ejemplos anteriores, se puede modificar más de una columna a la vez en una misma consulta. Y por supuesto, dependiendo de la condición que pongamos en el WHERE se pueden modificar desde cero registros (cuando la condición no se cumple en ninguno de los registros) a todos (cuando ponemos una condición que siempre se cumple o bien cuando no ponemos ninguna condición).
La última consulta es un claro ejemplo de esto último. Por supuesto que ejecutar este tipo de consultas es muy riesgoso y es poco probable que se deba realizar. En este apunte, se puso como ejemplo este caso para que se pueda apreciar qué pasaría si ejecutamos una consulta de UPDATE sin la cláusula WHERE.

Eliminar datos

La eliminación de datos definitiva de una tabla se realiza mediante la consultas del tipo DELETE. De esta manera nos aseguramos que los registros que deseemos borrar desaparezcan permanentemente de la base de datos.
Al igual que con las consultas de modificación, será muy importante que se especifique un criterio al momento de realizar la consulta (mediante la cláusula WHERE) de lo contrario estaremos borrando más registros de lo que queremos o peor aún, borrando la tabla completamente.

La forma general de las consultas de DELETE es la siguiente:

```
DELETE FROM <tabla> WHERE condicion
```

Ejemplos:

Eliminar el registro del alumno con legajo 9000

```
DELETE FROM alumnos WHERE legajo = 9000
```

Eliminar todos los registros de alumnos de sexo masculino que no tengan telefono o mail

```
DELETE FROM alumnos WHERE sexo = 'M' AND (email IS NULL OR telefono IS NULL)
```

Eliminar todos los registros de alumnos cuyo nombre comience con 'J' y su apellido termina con 'Z'

```
DELETE FROM alumnos WHERE nombre LIKE 'J%' AND apellido LIKE '%Z'
```

Eliminar todos los registros de alumnos

```
DELETE FROM alumnos
```

Como se puede observar en los ejemplos anteriores, las consultas de DELETE tienen gran similitud con las de UPDATE sólo que aquí no deben especificarse los campos a cambiar ya que se elimina todo el registro.
El uso de la cláusula WHERE es indispensable en este tipo de consultas. De lo contrario se obtendrán resultados indeseables, como el del último ejemplo que elimina por completo los registros de alumnos de la base de datos.