INSTITUTO DE EMPRENDEDORES

Definición de Datos (DDL)
BDSQL

Carrera: Computación e Informática

Semestre: III

Nombre de Unidad Didáctica. TALLER DE BASE DE DATOS

Indice

- Introducción: Bases de datos
- Modelo relacional
- SQL
  - Repaso de comandos principales
  - Lenguaje de definición de datos (DDL)
  - Lenguaje de manipulación (DML)
- Demostraciones
- Extensiones de SQL para el mundo SIG
- Problemas con el modelo relacional

¿Porque las bases de datos?

- Parece obvio hoy en día
- Tradicionalmente sistemas trabajaban a base de ficheros sueltos, y procedimientos sobre ellos
  - sistemas a medida de cada aplicación
  - Bdatos: separación de datos e su implementación (hardware/software)
  - Independencia
  - Protección (permite sistema multiusuario)
  - Flexibilidad (conectar la bdatos a todo)
  - Eficiencia (minimiza duplicidad de datos)
  - Integridad (minimiza errores lógicos)

Modelos de bases de datos

- Modelo jerárquico
  - estructura de árbol: relaciones 1:muchos
  - requiere duplicación de datos
- Modelo en red
  - permiten mejor relación entre los datos
  - todo conectado a todo
  - muy utilizado en aplicaciones COBOL (empresarial)
- Modelo relacional
  - modelo dominante hoy en día

Modelo relacional

- Dr Edgar (Ted) Codd, de la IBM
- 1970 “A relational model of data for large shared data banks” *Communications of the ACM 13(6).*
- Modelo muy simple, flexible hasta cierto punto
- Todo en tablas, con columnas y filas
- Operaciones para crear, borrar, modificar tablas
- Otras operaciones (álgebra relacional) para manipular (consultar) estas tablas...
- El modelo se caracteriza por tres elementos

Características del modelo

- Elemento estructural: forma de guardar datos
  - todo en tablas, y nada más que tablas
  - sin duplicar registros (filas, tuplas)
  - campos (columnas) con nombres únicos
  - entradas en un campo de solo un tipo
    - numérico (entero, real..), texto, fecha, etc.
  - todas las entradas serán datos atómicos
  - orden de filas/columnas no importa
  - valores nulos soportados (<> 0)
  - claves para crear relaciones (solo una es clave primaria)

Características

- Elemento de manipulación: que se puede hacer
  - Entrada: una o mas tablas
  - Salida: una tabla nueva
  - Codd define álgebra y cálculo relacional (el usuario no los vea)
  - En la práctica, solo son 3 operadores fundamentales:
    - SELECT: especificar “criterios de búsqueda” y crear una nueva tabla con solo los datos que buscábamos
    - PROJECT: copia un subconjunto de campos a una tabla nueva
    - JOIN: “pega” dos tablas para crear una nueva
  - Select y Join: operaciones críticas en el SIG vectorial

Características

- Elemento de integridad: control lógico
  - Integridad de entidades
    - garantiza que los campos clave tengan datos (no nulos) y que si existe un registro se puede localizar
  - Integridad referencial
    - mantiene intactas relaciones (referencias) de clave a clave
    - no puedes borrar un registro al que depende otra tabla
    - los dos campos clave deben ser del mismo tipo

SQL y el modelo relacional

- SQL no forma parte del modelo relacional
- Query-By-Example (QBE), otros lenguajes de consulta pueden aplicarse también al modelo
- SQL ha sido aceptado como el lenguaje *de facto*
- SQL aceptado por Codd, con matices
- Sirve como lenguaje completo: de definición (DDL) y de manipulación (DML) de datos según el modelo relacional
- Tiene una estructura “pseudo inglésa”
- Se utiliza como *lingua franca* entre sistemas

Repaso de comandos SQL

- DDL:
  - CREATE <tabla>
  - DROP <tabla>
- DML:
  - SELECT <columna(s) de datos>
  - FROM <tabla(s)>
  - WHERE <condición lógica>

Ejemplos del sintaxis SQL

```
– create table zona (
–   IdZona  smallint not null unique,
–   NomZona        char(30) not null unique,
–   Superf  smallint,
–   IdOfCD smallint not null
);
– create table tipo (
–   IdTipo   smallint not null unique,
–   DescTipo        char(30) not null unique
);
```

Mas ejemplos...

```
SELECT DISTINCT NomCons
FROM ofarea,relacion,ofcd,zona,parcela,construc
WHERE NomAr='Central'
AND ofarea.IdAr=relacion.IdAr
AND relacion.IdOfCD=zona.IdOfCD
AND zona.IdZona=parcela.IdZona
AND parcela.IdCons=construc.IdCons;
```

Repaso de comandos SQL

DDL:

CREATE <tabla>

DROP <tabla>

DML:

SELECT <columna(s) de datos>

FROM <tabla(s)>

WHERE <condición lógica>

Ejemplos del sintaxis SQL

```
create table zona (
  IdZona          smallint not null unique,
  NomZona         char(30) not null unique,
  Superf          smallint,
  IdOfCD          smallint not null
);
create table tipo (
  IdTiposmallint not null unique,
  DescTipo        char(30) not null unique
);
```

Mas ejemplos...

```
SELECT DISTINCT NomCons
FROM ofarea,relacion,ofcd,zona,parcela,construc
WHERE NomAr='Central'
AND ofarea.IdAr= relación.IdAr
AND relacion.IdOfCD= zona.IdOfCD
AND zona.IdZona= parcela.IdZona
AND parcela.IdCons= construc.IdCons;
```

Mas ejemplos...

```
SELECT NomAr,AVG( Superf),SUM (Superf)
FROM ofarea,relacion,zona
WHERE ofarea.IdAr= relacion.IdAr
AND relacion.IdOfCD= zona.IdOfCD
GROUP BY NomAr;
```

Relaciones

- Son BBDD relacionales, ¿no?
- Dividimos los datos entre varias tablas (específicas) para minimizar la duplicación de datos, y también las dependencias entre campos
  - proceso conocido como normalización
- Hay relaciones de 3 tipos entre atributos
  - 1:1, una persona tiene un DNI
  - 1:M, una persona tiene muchos amigos
  - M:N, una tienda tiene muchos clientes, cada uno de los cuales es cliente de muchas tiendas

Relaciones (2)

- El modelo relacional no permite relaciones M:N, por eso a veces hay que crear nuevas tablas (auxiliares) como “puentes” entre una tabla y otras
- Ejemplo de la Videoteca:
  - tabla “clientes” (cada cliente es único)
  - tabla “películas” (cada película es única)
  - Problema: ¿Como modelar el caso en que una película esta en manos de muchos clientes, y que cada cliente puede haber alquilado muchas películas?
- Solución: nueva tabla “movimientos”, con campos en común con “clientes” y “películas”

Claves

- Para enlazar tablas mediante un campo en común
- Claves primarias (campo único), como DNI en la tabla “clientes”
- Claves externas (foráneas), como DNI en la tabla “movimientos”

Diseño de la Base de Datos

- Cuales son las entidades (y sus atributos) de importancia
- Cuales son las relaciones entre ellas
- Creación de modelos E-A-R Luego diseñar una bdatos física de acuerdo con el modelo
- Este diseño no es una tarea trivial
- La explotación del SIG (consultas posibles) se basa en este diseño !!
- Rediseñar una base de datos a posteriori MUY caro !!

Transact-SQL: Introducción

- Transact-SQL es un lenguaje que sirve para la definición, tratamiento y control de los datos.

- Transact-SQL es el lenguaje de programación de MS-SQL Server.

INSTITUTO DE EMPRENDEDORES

Transact-SQL: Introducción

- El lenguaje de programación Transact-SQL
- Tipos de instrucciones de Transact-SQL
- Elementos de la sintaxis de Transact-SQL

El lenguaje de programación Transact-SQL

- Implementa el estándar ISO del nivel básico de la especificación ANSI SQL-92

  Los organismos ANSI (*American National Standards Institute*) e ISO (*International Standards Organization*) han definido estándares para SQL. Mediante Transact-SQL, Microsoft® SQL Server™ 2000 admite el nivel básico de implementación de SQL-92, el estándar SQL publicado por ANSI e ISO en 1992.

- Se pueden ejecutar desde cualquier producto que cumpla los requisitos básicos

Los elementos del lenguaje Transact-SQL que cumplen los requisitos de ANSI-SQL se pueden ejecutar desde cualquier producto que cumpla los requisitos básicos de ANSI-SQL.

- Incluye una funcionalidad ampliada

Transact-SQL incluye, además, varias extensiones que proporcionan una funcionalidad ampliada.

Tipos de instrucciones de Transact-SQL

- Una consulta es una petición que se hace para obtener datos almacenados en SQL Server.
- Todas las consultas presentan al usuario el conjunto de resultados de una instrucción SELECT.
- Un conjunto de resultados es una tabla que muestra los datos obtenidos mediante la instrucción SELECT. La tabla tiene filas y columnas.


INSTITUTO DE EMPRENDEDORES

Tipos de instrucciones de Transact-SQL

La escritura y ejecución de instrucciones de Transact-SQL es una de las formas en que se puede realizar una consulta en SQL Server.

Cuando escriba y ejecute instrucciones de Transact-SQL, utilizará:

- Instrucciones del Lenguaje de definición de datos (DDL)
  se utilizan para crear objetos en la base de datos.

- Instrucciones del Lenguaje de control de datos (DCL)
  se utilizan para determinar quién puede ver o modificar los datos.

- Instrucciones del Lenguaje de tratamiento de datos (DML)
  se utilizan para consultar y modificar los datos.

INSTITUTO DE EMPRENDEDORES

Instrucciones del Lenguaje de definición de datos (DDL)

- Definen los objetos de la base de datos

Las instrucciones de DDL definen la base de datos mediante la creación de bases de datos, tablas y tipos de datos definidos por el usuario. Las instrucciones de DDL se utilizan también para administrar los objetos de la base de datos. Algunas instrucciones de DDL son:

CREATE *nombreObjeto*
ALTER *nombreObjeto*
DROP *nombreObjeto*

- Deben tener los permisos adecuados

De forma predeterminada, sólo los miembros de la función **sysadmin**, **dbcreator**, **db_owner** o **db_ddladmin** pueden ejecutar instrucciones de DDL.

INSTITUTO DE EMPRENDEDORES

Instrucciones del Lenguaje de definición de datos (DDL)

EJEMPLO:

La secuencia de comandos siguiente crea una tabla llamada customer en la base de datos Northwind. Incluye las columnas cust_id, company, contact y phone.

```
USE northwind
CREATE TABLE customer
(cust_id int, company varchar(40),contact varchar(30), phone char(12) )
GO
```

INSTITUTO DE EMPRENDEDORES

Instrucciones del Lenguaje de control de datos (DCL)

- Establecer o cambiar los permisos

Las instrucciones de DCL se utilizan para cambiar los permisos asociados con un usuario o función de la base de datos. En la tabla siguiente se describen las instrucciones de DCL.

GRANT

Crea una entrada en el sistema de seguridad que permite a un usuario trabajar con datos o ejecutar ciertas instrucciones de Transact-SQL.

- DENY

Crea una entrada en el sistema de seguridad que deniega un permiso de una cuenta de seguridad e impide que el usuario, grupo o función herede el permiso a través de su pertenencia a grupos o funciones

- REVOKE

Quita un permiso concedido o denegado previamente.

Deben tener los permisos adecuados

De forma predeterminada, sólo los miembros de la función sysadmin, dbcreator, db_owner o db_securityadmin pueden ejecutar instrucciones DCL.

INSTITUTO DE EMPRENDEDORES

Instrucciones del Lenguaje de control de datos (DCL)

EJEMPLO:

En este ejemplo se concede a la función **public** el permiso para consultar la tabla **products**.

```
USE northwind
GRANT SELECT ON products TO public
GO
```

Instrucciones del Lenguaje de tratamiento datos (DML)

- Las instrucciones DML se utilizan para cambiar datos o recuperar información

Las instrucciones de DML funcionan con los datos de la base de datos. Mediante estas instrucciones puede cambiarlos o recuperar información.

Las instrucciones de DML incluyen:

- SELECT
- INSERT
- UPDATE
- DELETE

Deben tener los permisos adecuados

De forma predeterminada, sólo los miembros de la función **sysadmin**, **dbcreator**, **db_owner** o **db_datawriter** pueden ejecutar instrucciones DML.

INSTITUTO DE EMPRENDEDORES

Instrucciones del Lenguaje de tratamiento datos (DML)

EJEMPLO:

En este ejemplo se recupera el identificador de categoría, nombre de producto, identificador de producto y precio por unidad de los productos de la base de datos **Northwind**.

```
USE northwind
SELECT categoryid, productname, productid, unitprice
FROM products
GO
```

Elementos de la sintaxis de Transact-SQL

Las instrucciones de DML se crean a partir de varios elementos de la sintaxis de Transact-SQL. Entre estos elementos se encuentran los siguientes:

- Directivas de proceso por lotes
- Comentarios
- Identificadores
- Tipos de datos
- Variables
- Funciones del sistema
- Operadores
- Expresiones
- Elementos del lenguaje de control de flujo
- Palabras clave reservadas


¡GRACIAS!