Bases de Datos
Unidad 2
Modelo Relacional

Instructor: M.C. Luis Basto Díaz
Email: luisbasto@gmail.com

Modelo Relacional

- Álgebra Relacional
- Lenguaje SQL
- Integridad Referencial
- Normalización
  - Primera Forma Normal
  - Segunda Forma Normal
  - Tercera Forma Normal
  - Forma Normal de Boyce – Codd
  - Criterios de Normalización

Historia de SQL

- IBM produjo posteriormente el prototipo System R, basado en SEQUEL/2.
- Las raíces de SQL, sin embargo, están en SQUARE (Specifying Queries as Relational Expressions).
- En 1970, ORACLE apareció y fue probablemente el primer RDBMS comercial basado en SQL.
- En 1987, ISO y ANSI publicó un estándar inicial para SQL.
- En 1989, ISO publicó un apéndice que definió "características de mejora de la integridad".

3

Historia de SQL

- En 1992, se produjo la primera revisión mayor del estándar SQL, se refirió a esta como SQL/92.
- En 1999, SQL:1999 fue liberado con soporte para administración de datos orientado a objetos.
- En 2003, SQL:2003 fue liberado.

4

Componentes de SQL

- SQL incluye:
  - Lenguaje de definición de datos (LDD). Proporciona órdenes para la definición de esquemas de relación, borrado de relaciones, creación de índices y modificación de esquemas de relación.
  - Lenguaje de manipulación de datos (LMD). Incluye un lenguaje de consultas, basado tanto en el álgebra relacional como en el cálculo relacional de tuplas. Incluye también sentencias para insertar, borrar y modificar tuplas de la base de datos.

5

Componentes de SQL

- SQL incluye:
  - Definición de vistas. El LDD de SQL incluye sentencias para la definición de vistas.
  - Control de transacciones. SQL incluye sentencias para la especificación del comienzo y final de transacciones.
  - SQL incorporado y SQL dinámico. SQL dinámico e incorporado define cómo se pueden incorporar las instrucciones SQL en lenguajes de programación de propósito general, tales como C, C++, Java, PL/I, Cobol, Pascal y Fortran.

6

Componentes de SQL

- SQL incluye:
  - Integridad. El LDD de SQL incluye sentencias para la especificación de las restricciones de integridad que deben satisfacer los datos almacenados en la base de datos. Las actualizaciones que violen las restricciones de integridad se rechazan.
  - Autorización. El LDD de SQL incluye sentencias para especificar derechos de acceso para las relaciones y vistas.

7

Objetivos de SQL

- Crear la base de datos y las estructuras de relación.
- Realizar tareas básicas de gestión de datos, como inserción, modificación y borrado de los datos de las relaciones.
- Realizar consultas simples y complejas.
- Realizar las tareas anteriores con un esfuerzo mínimo por parte del usuario.
- Que sea portable.

8

SQL definición

- SQL (Structured Query Language, Lenguaje de Consulta Estructurado).
- SQL se trata de un lenguaje no procedural.
- Es de formato libre.
- Puede ser usado por diversos tipos de usuarios:
  - (DBA),
  - personal administrativo,
  - desarrolladores de aplicaciones y
  - muchos otros usuarios finales.

9

Escritura de comandos SQL

- Palabras reservadas
  - Son una parte fija del lenguaje.
  - Debe escribirse exactamente como se indica.
- Palabras definidas por el usuario
  - Son palabras compuestas por el usuario y representan nombres de diversos objetos de la base de datos, como tablas, columnas, vistas, índices, etc.

10

Recomendaciones

- Se usan letras mayúsculas para las palabras reservadas.
- Técnicas de sangrado y alineación
- Una barra vertical (|) indica una elección entre diversas alternativas. a|b|c
- Las {} indican un elemento obligatorio, {a}
- Los corchetes indican un elemento opcional [a]
- Los puntos suspensivos (…) indica repetición opcional de un elemento cero o más veces. {a|b}(,c…)

11

Tipos de datos SQL ISO

- Identificadores
  - Se utilizan para identificar objetos en la Base de datos, ejemplo: Nombres de tablas, nombres de vistas, columnas.
  - El estándar ISO permite letras mayúsculas A…Z, minúsculas a…z, dígitos 0…9 y el carácter _.
- Algunas restricciones
  - El identificador no puede tener más de 128 carac.
  - Cada identificador debe iniciar con una letra.
  - Los identificadores no pueden contener espacios.

12

Tipos de datos en SQL

Tipos de datos	Declaraciones
Booleano	BOOLEAN
Carácter	CHAR, VARCHAR
Numérico Exacto	NUMERIC, DECIMAL, INTEGER, SMALLINT
Numérico Aprox.	FLOAT, REAL, DOUBLE PRECISION
Fecha y Hora	DATE, TIME, TIMESTAMP
Intervalo	INTERVAL
Objetos	CHARACTER LARGE OBJ, BINARY LARGE OBJ

13

Definición de datos (DDL)

- El lenguaje de definición de datos permite crear y eliminar objetos de la BD, tales como: esquemas, dominios, tablas, vistas e índices.
- Las instrucciones principales son:
  - CREATE SCHEMA
  - CREATE TABLE
  - CREATE DOMAIN
  - CREATE VIEW
  - ALTER DOMAIN
  - ALTER TABLE
  - DROP SCHEMA
  - DROP DOMAIN
  - DROP TABLE
  - DROP VIEW

14

SCHEMA

- CREATE SCHEMA [nombre | authorization IdentificadorCreador]
- CREATE SCHEMA pruebasql AUTHORIZATION sa;
- DROP SCHEMA nombre [RESTRICT|CASCADE];

15

Creación de tablas

- Para crear tablas se deberá proveer el nombre de la tabla y los tipos de datos de las columnas:

```
CREATE TABLE
Table_name (
column_1 data_type,
column_2 data_type,
.
.
column_n data_type
)
```

```
USE Northwind
CREATE TABLE Drivers (
license VARCHAR(15),
firstname VARCHAR(30),
lastname VARCHAR(30)
)
```

16

Creación de tablas

- Para permitir/no permitir valores nulos en los campos, SQL provee de la sig. Sintaxis:

```
CREATE TABLE Cars (
    serial VARCHAR(200) NOT NULL,
    make VARCHAR(100) NOT NULL,
    model VARCHAR(100) NOT NULL,
    color VARCHAR(50) NULL
)
```

17

Modificando tablas

- Para modificar (eliminar/adicionar) una tabla previamente creada, se tiene los siguientes comandos:

```
ALTER TABLE Cars ADD mileage INT NULL
```

```
ALTER TABLE Cars DROP COLUMN mileage,color
```

18

Práctica

19

Condicionales

- El control de la integridad está compuesto por restricciones para proteger la BD frente a posibles errores o incoherencias.
  - Datos requeridos
  - Restricciones de dominio
  - Integridad de entidades
  - Integridad referencial
  - Restricciones generales

20

Condicionales

- Algunas columnas deben contener un valor válido.
- Los valores nulos son diferentes de los espacios en blanco o de los valores numéricos iguales a cero.
- El SQL ISO proporciona la palabra clave NOT NULL para esta restricción.
- Ejemplo: todos los empleados deben tener un cargo asociado.

21

Condicionales

- Toda columna debe tener un dominio.
- Ejemplo: Sexo de un empleado puede ser 'M' o 'F'.
- SQL ISO permite crear dominios con las cláusulas siguientes:
  - CREATE DOMAIN tiposex AS CHAR(4)
    CHECK(VALUE IN (SELECT col FROM tabla));

  Declaración:

  sex tiposex NOT NULL

22

Condicionales

- La clave principal de una tabla debe contener un valor unívoco y no nulo en cada fila.
- SQL ISO emplea las siguientes clásulas:
  - PRIMARY KEY(columna)
  - PRIMARY KEY(columna1, columna2)
- PRIMARY KEY se utiliza una sola vez para cada tabla.

23

Condicionales

- Una clave externa es una columna o conjunto de columnas que enlazan cada fila de la tabla hijo que contiene la tabla externa con la fila de la clave padre que contiene el valor correspondiente de clave candidata.
- Si la clave externa contiene un valor, dicho valor debe hacer referencia a una fila existente y válida dentro de la tabla padre.

24

Condicionales

- SQL ISO soporta las siguientes cláusulas:
  - FOREIGN KEY(columna) REFERENCES tabla.
- Las acciones referenciales son acciones que SQL ejecuta para evitar un error de integridad referencial.
  - CASCADE
  - SET NULL
  - SET DEFAULT
  - NO ACTION

25

Condicionales

- **Condicionales**
  - Las columnas de una tabla deben de contemplar ciertas condiciones o restricciones:
  - Ejemplo: el precio de un artículo debe de ser siempre mayor a 0.
  - Problemática: no existen tipos de datos que solo admitan números positivos.
  - En estos casos son usados los condicionales.

26

Condicionales

- *Condicional check*
  - Permite especificar que el valor de cierta columna debe de satisfacer una expresión booleana.

```
CREATE TABLE productos (
    producto_no integer,
    nombre text,
    precio numeric CHECK (precio > 0)
);
```

27

Condicionales

- *Condicional check*

```
CREATE TABLE productos (
    producto_no integer,
    nombre text,
    precio numeric,
    CHECK (precio > 0),
    descuento numeric,
    CHECK (descuento > 0),
    CHECK (precio > descuento)
);
```

28

Condicionales

- *Condicional not null*
  - Se aplica cuando una columna no permite "null" como valor, *los identificadores* son ejemplos claros.

```
CREATE TABLE productos (
    producto_no integer NOT NULL,
    nombre text NOT NULL,
    precio numeric
);
```

29

Condicionales

- *Combinación de condicionales*

```
CREATE TABLE productos (
    producto_no integer NOT NULL,
    nombre text NOT NULL,
    precio numeric NOT NULL CHECK (precio > 0)
);
```

30

Condicionales

- *Condicional UNIQUE*
  - Asegura que el dato contenido en una columna identifica a un renglón del resto existente en la tabla.

```
CREATE TABLE productos (
    producto_no integer UNIQUE,
    nombre text,
    precio numeric
);
```

31

Condicionales

- *Condicional UNIQUE*

```
CREATE TABLE productos (
    producto_no integer,
    nombre text,
    precio numeric,
    UNIQUE (producto_no)
);
```

```
CREATE TABLE
ejemplo (
    a integer,
    b integer,
    c integer,
    UNIQUE (a, c)
);
```

32

Condicionales

- *Condicional PRIMARY KEY*
  - Técnicamente este condicional es la combinación de los condicionales unique y not null.

```
CREATE TABLE productos (
    producto_no integer UNIQUE NOT NULL,
    nombre text,
    precio numeric
);
```

33

Condicionales

- *Condicional PRIMARY KEY*

```
CREATE TABLE productos (
    producto_no integer PRIMARY KEY,
    nombre text,
    precio numeric
);
CREATE TABLE ejemplo (
    a integer,
    b integer,
    c integer,
    PRIMARY KEY (a, c)
);
```

34

Condicionales

- *Condicional PRIMARY KEY*

```
CREATE TABLE productos (
        producto_no integer PRIMARY KEY,
        nombre text,
        precio numeric
);

CREATE TABLE ejemplo (
        a integer,
        b integer,
        c integer,
        PRIMARY KEY (a, c)
);
```

35

Condicionales

- *Condicional FOREIGN KEY*
  - Especifica que los valores en una columna (o grupo de columnas) deben de coincidir con los valores que aparecen en algún renglón de otra tabla.
  - Se dice que existe una **integridad referencial** entre dos tablas.

36

Condicionales

- Dada la tabla de *productos,* se define la tabla de *ordenes* la cual estará asociada a un producto (cardinalidad 1 a 1).
- ¿Cómo quedaría la definición de la tabla?

37

Condicionales

- Definición de la tabla *ordenes*:

```
CREATE TABLE ordenes (
    orden_id integer PRIMARY KEY,
    producto_no integer REFERENCES productos
(producto_no),
    cantidad integer
);
```

38

Condicionales

- *Condicional FOREIGN KEY*
  - Otra forma de definir el condicional:

```
CREATE TABLE t1 (
   a integer PRIMARY KEY,
   b integer,
   c integer,
   FOREIGN KEY (b, c) REFERENCES otra_tabla (c1, c2)
);
```

39

Condicionales

- ALTER TABLE
  - Adicionar una nueva columna de una tabla.
  - Eliminar una columna de una tabla.
  - Adicionar una nueva restricción a la tabla.
  - Eliminar una restricción a la tabla.
  - Asignar un default para una columna.
  - Eliminar un default para una columna.

40

Condicionales

- Cambiar la tabla Staff, removiendo el default 'Assistant' para la columna posición y asignar un default para la columna sexo a femenino ('F').

```
ALTER TABLE Staff
    ALTER position DROP DEFAULT;
ALTER TABLE Staff
    ALTER sex SET DEFAULT 'F';
```


41

Condicionales

- Remover la restricción de PropertyForRent, el cual indica que el empleado no está permitido manejar más de 100 propiedades a la vez. Adicionar una nueva columna a la tabla cliente.

```
ALTER TABLE PropertyForRent
    DROP CONSTRAINT StaffNotHandlingTooMuch;
ALTER TABLE Client
    ADD prefNoRooms PRooms;
```


42

Condicionales

DROP TABLE TableName [RESTRICT | CASCADE]

```
DROP TABLE PropertyForRent;
```

- Elimina la tabla y filas.
- RESTRICT: Si algún otro objeto depende de éste para su existencia, SQL no permite su eliminación.
- CASCADE: SQL elimina todos los objetos dependientes (y los objetos dependientes sobre estos objetos.