TRABAJO CONCEPTUAL SQL

MARIO JIMÉNEZ MARSET

ÍNDICE

1. ENUNCIADO - OBJETIVOS

En esta práctica se pedía realizar un trabajo de búsqueda, investigación y abstracción de información sobre SQL, cuyo eje sean cinco apartados:

- **Evolución de los Lenguajes de Programación.**
- **Sublenguajes del Ansi-SQL.**
- **Ámbitos de Trabajo y Métodos de Uso SQL: Estáticos y Dinámicos.**
- **Historia del SQL.**
- **Variaciones del Ansi-SQL.**

2. EVOLUCIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

El lenguaje de programación es un lenguaje formal diseñado para realizar **procesos** que pueden ser ejecutados por computadoras. Su uso puede ser para la **creación** de programas cuyo objetivo sea el de controlar el comportamiento físico y lógico de una máquina, para expresar **algoritmos** con precisión o como modo de comunicación humana.

Está formado por un conjunto de **símbolos** y **reglas** sintácticas/semánticas que definen su estructura y el significado de sus elementos y expresiones.

La "programación" en sí es el **proceso** por el cual se escribe, prueba, depura y compila el código fuente de un programa **informático**.

- El lenguaje **máquina** es una colección de dígitos binarios (bits) que la computadora lee o interpreta. Los lenguajes máquina son los únicos idiomas que las computadoras entienden.

  Estos, son casi imposibles de usar por los humanos, ya que consisten únicamente en **números**.

- El lenguaje **ensamblador** es el lenguaje utilizado para escribir programas informáticos de **bajo** nivel, siendo esta la representación más directa del código máquina específico para cada arquitectura de computadoras legible por un programador.

  Los programas de lenguaje ensamblador se traducen al lenguaje de máquina mediante un programa llamado ensamblador.

- Los lenguajes de alto nivel nos permiten escribir códigos de computadora usando instrucciones que se asemejan al lenguaje **humano** cotidiano, que luego se traducen al lenguaje de máquina para ser ejecutados.

  Los programas escritos en un lenguaje de alto nivel deben ser traducidos al lenguaje de máquina antes de que puedan ser **ejecutados**.

  Algunos lenguajes usan un compilador para realizar esta traducción y otros usan un **intérprete**.

Características de las **Generaciones** de los Lenguajes de Programación:

- **Primera Generación** (o 1GL): lenguajes de bajo nivel que son lenguaje de máquina.

- Los primeros ordenadores se programaban directamente en código binario.
- Cada modelo de ordenador tiene su propio código, por esa razón se llama lenguaje de máquina.
- La tecnología electrónica era a base de tubos de vacío.
- Se programaba en lenguaje de máquina.

- **Segunda Generación** (o 2GL): lenguajes de bajo nivel que generalmente consisten en lenguajes ensamblados.

- Los lenguajes simbólicos (propios de la máquina), simplifican la escritura de las instrucciones y las hacen más legibles.
- Aparición del circuito integrado (chips).
- Disminución del tamaño.
- Disminución del consumo y de la producción del calor.
- Su fiabilidad alcanza metas inimaginables con los efímeros tubos de vacío.

- **Tercera Generación** (o 3GL): lenguajes de alto nivel.

- Los lenguajes de alto nivel sustituyen las instrucciones simbólicas por códigos independientes de la máquina, parecidas al lenguaje humano o al de las Matemáticas.
- Circuito integrado desarrollado en 1958 por Jack Kilby.
- Menor consumo de energía.
- Circuito integrado, miniaturización y reunión de centenares de elementos en una placa de silicio o (chip).
- Aumento de fiabilidad y flexibilidad.

- **Cuarta Generación** (o 4GL): idiomas que consisten en declaraciones similares a las declaraciones en un lenguaje humano. Se usan comúnmente en la programación de bases de datos y scripts.

- Ciertas herramientas que permiten construir aplicaciones sencillas combinando piezas prefabricadas.
- Estas herramientas no son lenguajes.
- Se minimizan los circuitos, aumenta la capacidad de almacenamiento.
- Reducen el tiempo de respuesta.

El lenguaje "**SQL**" se clasifica dentro de esta **cuarta** generación, ya que es usado en aplicaciones de gestión y manejo de bases de datos.
Fue creado por **IBM** con el fin de administrar, proteger y recuperar los datos de sistemas de gestión en información. Es muy usado en **ciberseguridad**.

Se caracteriza por ser muy fácil de aprender, ya que presenta una forma estandarizada de interactuar con la base de datos. Es uno de los lenguajes que más se aproxima a la lengua anglosajona en términos de sintaxis.

- **Quinta Generación** (o 5GL): lenguajes de programación que contienen herramientas visuales para ayudar a desarrollar un programa.

- Lenguajes de la inteligencia artificial.
- Mayor miniaturización de los elementos.
- Aumenta la capacidad de memoria.
- Multiprocesador (procesadores interconectados).
- Mayor velocidad.

3. SUBLENGUAJES DEL ANSI-SQL

El SQL es un lenguaje diseñado para administrar y recuperar **información** de sistemas de gestión de bases de datos relacionales.

Una de sus principales características es el manejo del **álgebra** y el **cálculo relacional** para efectuar **consultas** con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas.

Basado en el álgebra relacional y en el cálculo relacional, SQL consiste en un lenguaje de definición, manipulación y control de datos. Incluye la **inserción** de datos, consultas, actualizaciones y borrado, creación y modificación de esquemas, control de acceso a datos…

SQL pasó a ser el estándar del Instituto Nacional Estadounidense de Estándares (**ANSI**) en 1986.

La programación SQL se puede usar para **compartir** y **administrar** datos (la información organizada en tablas que se encuentra en los sistemas de administración de bases de datos relacionales).

Mediante este lenguaje se puede:
- Consultar, actualizar y reorganizar **datos**.
- Crear y modificar la estructura de un sistema de **base** de datos.
- Controlar el **acceso** a sus datos.

Los sistemas de gestión de bases de datos con soporte SQL más utilizados son:
- **DB2**.
- **HSQL**.
- **InterBase**.
- **Oracle**.
- **MySQL**.
- **PostgreSQL**.
- **SQL Lite**.

Del SQL se derivan sublenguajes los cuales son utilizados para la manipulación, definición y control de la información o estructura de una base de datos relacional.

Los **sublenguajes** son:

- **DDL** (Data Definition Language):
  Es el sublenguaje que se encarga de la modificación de la estructura de los objetos de la base de datos. Incluye órdenes para modificar, borrar o definir tablas, vistas, base de datos, procedimientos almacenados o funciones.

  Las órdenes son:

  - **Create**.
  - **Alter**.
  - **Drop**.
  - **Truncate**.

- **TCL** (Transaction Control Language):

Es un subconjunto de SQL que se utiliza para controlar el procesamiento de transacciones en una base de datos. Una transacción es una unidad lógica de trabajo que comprende una o más sentencias SQL.

Son grupo de sentencias que se encuentran en el sublenguaje DML.

Los comandos que posee este sublenguaje son:

- **Commit**.
- **Rollback**.
- **Savepoint**.

- **DCL** (Data Control Language):
  Es un sublenguaje que incluye una serie de comandos que permiten la administración del control de acceso de datos contenidos en la base de datos.

  Los comandos incluidos en este sublenguaje son:

  - **Grant**.
  - **Revoke**.

- **DML** (Data Manipulation Language):
  Permite a los usuarios llevar a cabo las tareas de consulta o manipulación de datos. Recoge todas las operaciones de intercambio de datos entre tablas.

  Las operaciones se dividen en:

- Consultas → recuperación de información.
- Tratamiento de Datos → insertar, actualizar y eliminar.

Los comandos utilizados en este sublenguaje son:

- **Select**.
- **Insert**.
- **Update**.
- **Delete**.

4. ÁMBITOS DE TRABAJO Y MÉTODOS DE USO SQL: ESTÁTICOS Y DINÁMICOS

Las **sentencias** SQL pertenecen a dos categorías principales: **DDL** y **DML**. Estos dos clasifican las sentencias de lenguaje SQL en función de su cometido.

La diferencia principal es que DDL crea **objetos** en la base de datos y sus efectos se pueden ver en el diccionario de la base de datos; sin embargo, DML permite consultar, insertar, modificar y eliminar la información almacenada en los objetos de la base de datos.

Cuando se ejecutan las sentencias DDL de SQL, el Sistema Gestor de la Base de Datos confirma la **transacción** actual antes y después de cada una de las sentencias DDL. En cambio, las sentencias DML no llevan implícito el "**commit**" y se pueden deshacer.

En ocasiones se puede presentar el problema de **mezclar** las sentencias DML con las DDL, ya que estas últimas pueden confirmar las primeras de manera involuntaria e implícita.

El SQL en sí se utiliza en la industria de la salud, el retail, la educación o las telecomunicaciones. Incluso tiene aplicaciones en la industria de la defensa.

El SQL **estático** son declaraciones en una aplicación que no cambian en **tiempo** de **ejecución** (cuando se escribe el código fuente) y, por lo tanto, pueden codificarse en la aplicación. Se denomina estático porque las sentencias SQL del programa no cambian cada vez que se ejecuta.

Se utiliza generalmente en la aplicación de base de datos SQL incorporada.

El gestor de bases de datos **DB2** debe procesar la fuente mediante un precompilador de SQL antes de que pueda compilarse y ejecutarse.

Durante este proceso, el precompilador SQL evalúa las **referencias** a tablas, columnas y **declara** los tipos de datos de todas las variables del lenguaje principal y determina qué métodos de **conversión** de datos deben usarse cuando los datos se mueven hacia la base de datos y desde ella misma.

Las sentencias son limitadas porque el precompilador debe conocer su **formato** de antemano y porque solo pueden trabajar con variables del lenguaje principal.

El SQL estático se puede ejecutar desde varias **interfaces**:

- Aplicaciones SQL integradas.
- Rutinas SQL integradas.
- Aplicaciones SQLJ.
- Rutinas SQLJ.
- Rutinas SQL.

En SQL estático, el procedimiento de acceso a la base de datos está predeterminado en la declaración de SQL estático y lo realiza el **preprocesador** cuando el usuario no puede ejecutar consultas en tiempo de ejecución.

Todos los procesos (incluidos el análisis, validación, optimización y generación del plan de acceso a la aplicación), se realizan en el momento de la **compilación**.

Declaraciones como EJECUTAR INMEDIATO o EJECUTAR PREPARAR no se utilizan.

Las sentencias de SQL estático se compilan en tiempo de compilación.
El SQL estático se utiliza en el caso de datos distribuidos uniformemente.

Las sentencias de SQL estático son más **rápidas** y **eficientes**.

El SQL estático es menos **flexible**.

El SQL **dinámico** se usa generalmente para enviar declaraciones SQL al administrador de la base de datos desde **interfaces gráficas** de usuario de creación de consultas interactivas y procesadores de línea de comandos SQL, así como desde **aplicaciones** donde no se conoce la estructura completa de las consultas en el momento de la compilación de la aplicación y la API de programación admite SQL dinámico.

El SQL dinámico puede construirse en **tiempo de ejecución** y colocarse en un host de cadena. Luego se envían al sistema de gestión de la base de datos para su **procesamiento**. Con SQL dinámico, puede crear aplicaciones **flexibles** de uso más general utilizando SQL porque el texto completo de una declaración SQL puede ser desconocido en la compilación.

Es más **lento** que el SQL estático puesto que el sistema de administración de bases de datos tiene que crear un plan de acceso en tiempo de ejecución para las sentencias de SQL dinámico.

Sin embargo, cuando se compila un programa que contiene sentencias de SQL dinámico, las sentencias de SQL dinámico **no** se **eliminan** del programa, como en el SQL estático, sino que se reemplazan por una llamada de función que pasa la sentencia al sistema de administración de bases de datos.

Se puede utilizar desde varias **interfaces**:

- Ventanas de comandos DB2.
- Procesador de línea de comandos DB2.
- Interfaces GUI de DB2 interactivas.
- Aplicaciones y rutinas externas que emplean API que admiten SQL dinámico.

En SQL dinámico, la forma en que se accede a una base de datos se determina en tiempo de ejecución y el usuario también puede ejecutar **instrucciones** de lenguaje de consulta estructurado en tiempo de ejecución.

Todos los procesos, incluido el análisis, la validación, la optimización y la generación del plan de acceso a la aplicación (forma binaria de consultas SQL) se realizan en tiempo de ejecución.

Se utilizan declaraciones como **EJECUTAR INMEDIATO**, **EJECUTAR PREPARAR**.

Las sentencias de SQL dinámico se compilan en tiempo de ejecución.

El SQL dinámico se utiliza en el caso de datos distribuidos de manera no uniforme.

Las sentencias de SQL dinámico son menos **eficientes**.

El SQL dinámico es muy **flexible**.

5. HISTORIA DEL SQL

En 1970, Frank Codd propone el **modelo relacional** y asociado a este un sublenguaje de acceso a los datos basado en el cálculo de predicados.

Los laboratorios de IBM definieron el lenguaje **SEQUEL** (Structured English Query Language) que más tarde fue ampliamente implementado por el sistema de gestión de bases de datos experimental **System R**, desarrollado en 1977 también por IBM.

Sin embargo, Oracle fue quien lo introdujo por primera vez en 1979 como un producto comercial.

El SEQUEL terminó siendo el predecesor de SQL, que es una versión evolucionada del primero. SQL pasa a ser el lenguaje por excelencia de los diversos sistemas de gestión de bases de datos relacionales surgidos en los años siguientes y fue estandarizado en 1986 por el **ANSI**, dando lugar a la primera versión estándar de este lenguaje "**SQL-86**" o "SQL1". Este estándar es también adoptado por **ISO**.

Sin embargo, este primer estándar no cubría todas las necesidades de los desarrolladores e incluía funcionalidades de definición de almacenamiento que se consideró suprimirlas. Así que, en 1992, se lanzó un nuevo estándar ampliado y revisado de SQL llamado "**SQL-92**" o "**SQL2**".

Actualmente, SQL es el estándar de la inmensa mayoría de los SGBD comerciales. El soporte al estándar SQL-92 es general y muy amplio.

Versiones a lo largo del tiempo:
- 1986: **SQL-86** → primera publicación hecha por ANSI. Confirmada por la Organización Internacional de Normalización en 1987.
- 1989: **SQL-89** → revisión menor.
- 1992: **SQL-92** → revisión mayor.
- 1999: **SQL:1999** → se agregaron expresiones regulares, consultas recursivas, triggers y algunas características orientadas a objetos.
- 2003: **SQL:2003** → define las maneras en las cuales SQL se puede utilizar conjuntamente con XML. Define maneras de importar y guardar datos XML en una base de datos SQL, manipulándolos dentro de la base de datos y publicando el XML y los datos SQL convencionales en forma de XML. Proporciona facilidades que permiten a las aplicaciones integrar dentro de su código SQL el uso de XQuery, lenguaje de consulta XML publicado por el W3C para acceso concurrente a datos ordinarios SQL y documentos XML.
- 2008: **SQL:2008** → permite el uso de la cláusula ORDER BY fuera de las definiciones de los cursores. Incluye los disparadores del tipo INSTEAD OF. Añade la sentencia TRUNCATE.
- 2011: **SQL:2011** → mejoras en las funciones de ventana y de la cláusula FETCH. Datos temporales (PERIOD FOR).
- 2016: **SQL:2016** → permite búsquedas de patrones, funciones de tabla polimórficas y compatibilidad con los ficheros JSON.

6. VARIACIONES DEL ANSI-SQL

Existen distintas variaciones del Ansi-SQL, como el **PL/SQL** o **Transact-SQL**.

El PL/SQL es el lenguaje **procedimental** de Oracle, lanzado en 1992. Es una extensión de SQL. El interés del lenguaje PS/SQL está en poder mezclar la **potencia** de instrucciones SQL con la **flexibilidad** de un lenguaje procedimental en un mismo programa.

Estos programas pueden ser ejecutados directamente por las herramientas de ORACLE (**bloques anónimos**) o a partir de objetos de la base de datos (procedimientos almacenados y paquetes).

Las ventajas del lenguaje PL/SQL son diversas:

- **Integración de SQL**: se pueden utilizar las instrucciones DML, las de control de transacciones y las funciones SQL prácticamente con la misma sintaxis.

- **Procesamiento procedimental**: la gestión de variables y las estructuras de control (condiciones y bucles) incrementan las posibilidades de gestión de los datos.

- **Funcionalidades suplementarias**: la gestión de cursores y el tratamiento de errores ofrecen nuevas posibilidades de procesamiento.

- **Mejora del rendimiento**: se pueden agrupar varias instrucciones en una misma unidad (bloque) que sólo dará lugar a un "acceso" a la base de datos (en lugar de un acceso por cada instrucción).

- **Integración en los productos de ORACLE**: los bloques o procedimientos PL/SQL son compilados y ejecutados por el "motor" de PL/SQL. Este motor está integrado en el motor de la base de datos, así como en determinadas herramientas...

Este lenguaje incluye nuevas características como el manejo de **variables**, estructuras **modulares**, estructuras de control de **flujo** y toma de **decisiones**, control de **excepciones**…

Es un lenguaje incorporado en el servidor de la base de datos y en las herramientas de Oracle (como Forms, Reports…).

Transact-SQL (T-SQL) es una extensión al SQL de Microsoft y Sybase.
SQL, que frecuentemente se dice ser un lenguaje de búsquedas estructurado, es un lenguaje de cómputo estandarizado, desarrollado originalmente por IBM para realizar **búsquedas**, **alterar** y **definir** bases de datos relacionales utilizando sentencias declarativas.

T-SQL expande el estándar de SQL para incluir programación procedimental, variables locales, varias funciones de soporte para **procesamiento** de strings, procesamiento de fechas, matemáticas…, y cambios a las sentencias **DELETE** y **UPDATE**. Estas características adicionales hacen de T-SQL un lenguaje que cumple con las características de un autómata de Turing.

TRANSACT-SQL es un lenguaje muy **potente** que nos permite definir casi cualquier tarea que queramos efectuar sobre la base de datos; incluye características propias de cualquier lenguaje de programación, características que nos permiten definir la lógica necesaria para el tratamiento de la información:

- Tipos de datos.
- Definición de variables.
- Estructuras de control de flujo.
- Gestión de excepciones.
- Funciones predefinidas.

Este potente lenguaje no permite crear interfaces de usuario o aplicaciones ejecutables (o sino elementos que en algún momento llegarán al servidor de datos y serán ejecutados).

Debido a estas restricciones se emplea generalmente para crear **procedimientos almacenados**, **triggers** y **funciones** de usuario.

Puede ser utilizado como cualquier SQL como un lenguaje utilizado en **aplicaciones** desarrolladas en otros lenguajes de programación como Visual Basic, C, Java…, además de los lenguajes incluidos en la plataforma **.NET**.

También lo podremos ejecutar directamente de manera interactiva.
Por ejemplo desde el editor de consultas de **SSMS** (SQL Server Management Studio).

7. WEBGRAFÍA

- https://conogasi.org/articulos/lenguaje-de-programacion/
- https://lacayo19.angelfire.com/lenguajes.htm
- https://www.ecured.cu/Lenguaje_ensamblador
- https://www.edix.com/es/instituto/lenguajes-de-programacion/
- https://mauricioaguilar1825.wordpress.com/sql/
- https://lasdiferencias.com/diferencias-sql-dinamico-estatico/
- https://www.infor.uva.es/~jvegas/cursos/bd/sqlplus/sqlplus.html#tiposSentencias
- https://es.wikipedia.org/wiki/SQL
- https://www.ediciones-eni.com/open/mediabook.aspx?idR=8212a1928eb9332b9ce4190c85361782
- https://es.wikipedia.org/wiki/PL/SQL
- https://es.wikipedia.org/wiki/Transact-SQL