|  |
| --- |
|  |
|  |  |



# C1 DESARROLLO DE CASO

**TERCER PARCIAL**

**INGENIERIA EN SISTEMAS**

**BASE DE DATOS 2**

# C1 DESARROLLO DE CASO

**TERCER PARCIAL**

**INGENIERIA EN SISTEMAS**

**BASE DE DATOS 2**

**UNIVERSIDAD NACIONAL AUTÓNOMA DE HONDURAS**

**CENTRO UNIVERSITARIO REGIONAL DEL LITORAL PACÍFICO**

**(UNAH-CURLP)**

**Bases de Datos II**

**(IS601)**

**Docente: Ing. Oscar Omar Pineda, MSc**

**Integrantes:**

**Ernesto Noe Moncada 20202300046**

**Adair Humberto Flores 20192300102**

**Josué Daniel Henríquez 20202300042**

**Esdras Rigoberto Castillo 20172300218**

**Francisco Josafat Paz Flores 20212300157**

**Kenis Noe Osorto Reyes 20212300177**

**Choluteca abril 2024**

# CONTENIDO

[C1 DESARROLLO DE CASO 1](#_Toc163994371)

[1 CONTENIDO 3](#_Toc163994372)

[2 TABLA DE ILUSTRACIONES 4](#_Toc163994373)

[3 INTRODUCCIÓN 5](#_Toc163994374)

[4 OBJETIVOS 6](#_Toc163994375)

[4.1 General 6](#_Toc163994376)

[4.2 Específicos 6](#_Toc163994377)

# TABLA DE ILUSTRACIONES

# INTRODUCCIÓN

Desde su lanzamiento PostgreSQL ha evolucionado como un sistema de gestión de bases de datos relacional de código abierto, destacándose por su flexibilidad y su habilidad para adaptarse a una diversidad de entornos y necesidades de datos. Gracias a su arquitectura altamente modular y una rica variedad de características, PostgreSQL sobresale en la gestión tanto de transacciones críticas como en el análisis de datos complejos.

La comunidad activa que rodea a PostgreSQL asegura un proceso de desarrollo sólido y continuo, lo que garantiza que la base de datos se mantenga relevante y actualizada frente a las demandas cambiantes del mundo de la gestión de datos. Esta colaboración continua también se refleja en el extenso soporte ofrecido, que garantiza la confiabilidad y la estabilidad de PostgreSQL a largo plazo.

PostgreSQL no solo se destaca por proporcionar una base sólida para el modelado de datos y la aplicación de restricciones de integridad referencial, sino que también impulsa una cultura de colaboración y contribución. Este enfoque promueve el desarrollo de un ecosistema de software dinámico y accesible. En proyectos como la programación mensual del canal de TV por cable CINEX con el propósito de publicar la programación en Internet, PostgreSQL se convierte en una opción confiable, permitiendo el diseño de una estructura de base de datos eficiente y segura, adaptable a las necesidades específicas del negocio.

# OBJETIVOS

## General

* Diseñar e implementar una base de datos normalizada en PostgreSQL para la programación mensual del canal de TV por cable CINEX, con el propósito de publicar la programación en Internet, asegurando una gestión eficiente y precisa de la información relacionada con los programas, horarios, géneros y fechas de emisión.

## Específicos

* Ingresar datos de prueba que representen diversos escenarios de programación, garantizando su conformidad con las restricciones y relaciones establecidas.
* Realizar consultas de validación para asegurar que la integridad referencial se mantenga intacta, incluso con datos simulados de muestra.

# PostGreSQL

## ¿Qué es PostgreSQL?

PostgreSQL es un potente sistema de base de datos relacional de objetos de código abierto que utiliza y amplía el lenguaje SQL combinado con muchas funciones que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas. Los orígenes de PostgreSQL se remontan a 1986 como parte del proyecto POSTGRES de la Universidad de California en Berkeley y cuenta con más de 35 años de desarrollo activo en la plataforma central (PostgreSQL, 2024).

PostgreSQL se ha ganado una sólida reputación por su arquitectura probada, confiabilidad, integridad de datos, conjunto de características robustas, extensibilidad y la dedicación de la comunidad de código abierto detrás del software para ofrecer constantemente soluciones innovadoras y de alto rendimiento. PostgreSQL se ejecuta en los principales sistemas operativos, ha sido compatible con ACID desde 2001 y tiene potentes complementos como el popular extensor de base de datos geoespacial PostGIS. No es de extrañar que PostgreSQL se haya convertido en la base de datos relacional de código abierto elegida por muchas personas y organizaciones (PostgreSQL, 2024).

Comenzar a usar PostgreSQL nunca ha sido tan fácil: elija un proyecto que desee crear y deje que PostgreSQL almacene sus datos de forma segura y sólida (PostgreSQL, 2024).

### Características:

#### Tipos de datos

* Primitivas: Entero, Numérico, Cadena, Booleano
* Estructurado: Fecha/Hora, Matriz, Rango/Multirango, UUID
* Documento: JSON/JSONB, XML, Clave-valor (Hstore)
* Geometría: Punto, Línea, Círculo, Polígono
* Personalizaciones: Compuesto, Tipos personalizados

#### Integridad de los datos

* ÚNICO, NO NULO
* Claves primarias
* Claves foráneas
* Restricciones de exclusión
* Bloqueos explícitos, bloqueos de asesoramiento

#### Simultaneidad, rendimiento

* Indexación: Árbol B, Multicolumna, Expresiones, Parcial
* Indexación avanzada: GiST, SP-Gist, KNN Gist, GIN, BRIN, índices de cobertura, filtros de Bloom
* Planificador/optimizador de consultas sofisticado, análisis de solo índice, estadísticas de varias columnas
* Transacciones, transacciones anidadas (a través de puntos de guardado)
* Control de simultaneidad de varias versiones (MVCC)
* Paralelización de consultas de lectura y creación de índices de árbol B
* Particionamiento de tablas
* Todos los niveles de aislamiento de transacciones definidos en el estándar SQL, incluido Serializable
* Compilación Just-In-Time (JIT) de expresiones

#### Confiabilidad, recuperación ante desastres

* Registro de escritura anticipada (WAL)
* Replicación: asíncrona, síncrona, lógica
* Recuperación a un momento dado (PITR), en espera activa
* Espacios de tablas

#### Seguridad

* Autenticación: GSSAPI, SSPI, LDAP, SCRAM-SHA-256, certificado y más
* Robusto sistema de control de acceso
* Seguridad a nivel de columna y fila
* Autenticación multifactor con certificados y un método adicional

#### Extensibilidad

* Funciones y procedimientos almacenados
* Lenguajes procedimentales: PL/pgSQL, Perl, Python y Tcl. Hay otros lenguajes disponibles a través de extensiones, por ejemplo, Java, JavaScript (V8), R, Lua y Rust
* Constructores SQL/JSON y expresiones de ruta de acceso
* Contenedores de datos externos: conéctese a otras bases de datos o flujos con una interfaz SQL estándar
* Interfaz de almacenamiento personalizable para tablas
* Muchas extensiones que proporcionan funcionalidad adicional, incluyendo PostGIS

#### Internacionalización, Búsqueda de texto

* Compatibilidad con conjuntos de caracteres internacionales, por ejemplo, a través de intercalaciones de ICU
* Intercalaciones que no distinguen entre mayúsculas y minúsculas ni acentos
* Búsqueda de texto completo

(PostgreSQL, 2024)

## Breve historia de PostgreSQL

El sistema de gestión de bases de datos relacionales de objetos, ahora conocido como PostgreSQL, se deriva del paquete POSTGRES escrito en la Universidad de California en Berkeley. Con décadas de desarrollo a sus espaldas, PostgreSQL es ahora la base de datos de código abierto más avanzada disponible en cualquier lugar.

## El proyecto POSTGRES de Berkeley

El proyecto POSTGRES, dirigido por el profesor Michael Stonebraker, fue patrocinado por la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA), la Oficina de Investigación del Ejército (ARO), la Fundación Nacional de Ciencias (NSF) y ESL, Inc. La implementación de POSTGRES comenzó en 1986. Los conceptos iniciales del sistema se presentaron en [ston86] y la definición del modelo de datos inicial apareció en [rowe87]. El diseño del sistema de reglas en ese momento fue descrito en [ston87a]. La lógica y la arquitectura del administrador de almacenamiento se detallaron en [ston87b].

POSTGRES ha sufrido varios lanzamientos importantes desde entonces. El primer sistema "demoware" entró en funcionamiento en 1987 y se mostró en la Conferencia ACM-SIGMOD de 1988. La versión 1, descrita en [ston90a], fue lanzada a unos pocos usuarios externos en junio de 1989. En respuesta a una crítica del primer sistema de reglas ([ston89]), el sistema de reglas fue rediseñado ([ston90b]), y la versión 2 fue lanzada en junio de 1990 con el nuevo sistema de reglas. La versión 3 apareció en 1991 y agregó soporte para múltiples administradores de almacenamiento, un ejecutor de consultas mejorado y un sistema de reglas reescrito. En su mayor parte, las versiones posteriores hasta Postgres95 (ver más abajo) se centraron en la portabilidad y la confiabilidad.

POSTGRES se ha utilizado para implementar muchas aplicaciones diferentes de investigación y producción. Estos incluyen: un sistema de análisis de datos financieros, un paquete de monitoreo del rendimiento del motor a reacción, una base de datos de seguimiento de asteroides, una base de datos de información médica y varios sistemas de información geográfica. POSTGRES también se ha utilizado como herramienta educativa en varias universidades. Finalmente, Illustra Information Technologies (más tarde fusionada con Informix, que ahora es propiedad de IBM) recogió el código y lo comercializó. A finales de 1992, POSTGRES se convirtió en el principal gestor de datos para el proyecto de computación científica Sequoia 2000.

El tamaño de la comunidad de usuarios externos casi se duplicó en 1993. Se hizo cada vez más evidente que el mantenimiento del código y el soporte del prototipo estaba ocupando grandes cantidades de tiempo que deberían haberse dedicado a la investigación de bases de datos. En un esfuerzo por reducir esta carga de soporte, el proyecto Berkeley POSTGRES finalizó oficialmente con la versión 4.2.

## Postgres95

En 1994, Andrew Yu y Jolly Chen añadieron un intérprete de lenguaje SQL a POSTGRES. Bajo un nuevo nombre, Postgres95 fue lanzado posteriormente a la web para encontrar su propio camino en el mundo como un descendiente de código abierto del código original de POSTGRES Berkeley.

El código Postgres95 era completamente ANSI C y se había recortado su tamaño en un 25%. Muchos cambios internos mejoraron el rendimiento y la capacidad de mantenimiento. La versión 1.0.x de Postgres95 se ejecutó entre un 30 y un 50 % más rápido en Wisconsin Benchmark en comparación con POSTGRES, versión 4.2. Aparte de las correcciones de errores, las siguientes fueron las principales mejoras:

* El lenguaje de consulta PostQUEL fue reemplazado por SQL (implementado en el servidor). (La biblioteca de interfaz libpq lleva el nombre de PostQUEL). Las subconsultas no eran compatibles hasta PostgreSQL (ver más abajo), pero podían ser imitadas en Postgres95 con funciones SQL definidas por el usuario. Se volvieron a implementar las funciones agregadas. También se agregó soporte para la cláusula de consulta GROUP BY
* Se proporcionó un nuevo programa (psql) para consultas SQL interactivas, que utilizaba GNU Readline. Esto reemplazó en gran medida al antiguo programa de monitores.
* Una nueva biblioteca front-end, , admite clientes basados en Tcl. Un shell de ejemplo, , proporcionó nuevos comandos Tcl para interconectar los programas Tcl con el servidor Postgres95.libpgtclpgtclsh
* Se revisó la interfaz de objetos grandes. La inversión de objetos grandes era el único mecanismo para almacenar objetos grandes. (Se ha eliminado el sistema de archivos de inversión).
* Se ha eliminado el sistema de reglas de nivel de instancia. Las reglas seguían estando disponibles como reglas de reescritura.
* Con el código fuente se distribuyó un breve tutorial que introducía las características regulares de SQL, así como las de Postgres95
* Se utilizó GNU make (en lugar de BSD make) para la compilación. Además, Postgres95 se podía compilar con un GCC sin parches (se corrigió la alineación de datos de los dobles).

## PostgreSQL

En 1996, quedó claro que el nombre "Postgres95" no resistiría la prueba del tiempo. Elegimos un nuevo nombre, PostgreSQL, para reflejar la relación entre el POSTGRES original y las versiones más recientes con capacidad SQL. Al mismo tiempo, establecimos la numeración de la versión para que comience en 6.0, volviendo a colocar los números en la secuencia originalmente iniciada por el proyecto POSTGRES de Berkeley.

Mucha gente sigue refiriéndose a PostgreSQL como "Postgres" (ahora rara vez en mayúsculas) debido a la tradición o porque es más fácil de pronunciar. Este uso es ampliamente aceptado como apodo o alias.

Durante el desarrollo de Postgres95, el énfasis estuvo en identificar y comprender los problemas existentes en el código del servidor. Con PostgreSQL, el énfasis se ha desplazado hacia el aumento de características y capacidades, aunque el trabajo continúa en todas las áreas.

Los detalles sobre lo que ha sucedido en PostgreSQL desde entonces se pueden encontrar en el Apéndice E.

## Documentación

### Creación de una base de datos

La primera prueba para ver si puede acceder al servidor de bases de datos es intentar crear una base de datos. Un servidor PostgreSQL en ejecución puede administrar muchas bases de datos. Normalmente, se utiliza una base de datos independiente para cada proyecto o para cada usuario.

Para crear una nueva base de datos, en este ejemplo denominada mydb, utilice el siguiente comando:

$ createdb mydb

También puede crear bases de datos con otros nombres. PostgreSQL le permite crear cualquier número de bases de datos en un sitio determinado. Los nombres de las bases de datos deben tener un primer carácter alfabético y están limitados a 63 bytes de longitud. Una opción conveniente es crear una base de datos con el mismo nombre que su nombre de usuario actual. Muchas herramientas asumen que el nombre de la base de datos es el predeterminado, por lo que puede ahorrarle algo de escritura. Para crear esa base de datos, simplemente escriba:

$ createdb

Si ya no desea utilizar su base de datos, puede eliminarla. Por ejemplo, si usted es el propietario (creador) de la base de datos, puede destruirla mediante el siguiente comando:mydb

$ dropdb mydb

(Para este comando, el nombre de la base de datos no se establece de forma predeterminada en el nombre de la cuenta de usuario. Siempre es necesario especificarlo). Esta acción elimina físicamente todos los archivos asociados con la base de datos y no se puede deshacer, por lo que esto solo debe hacerse con mucha previsión.