Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

upiita-ipn

Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas

Alumno
Rivera Ortiz Raúl Alejandro
Solano Castrejón Eric

Unidad de Aprendizaje:
Bases de Datos Distribuidas

Grupo
3TM3

Profesor
De La Cruz Sosa Carlos

Actividad
Arquitecturas de BDD

9 de marzo de 2022

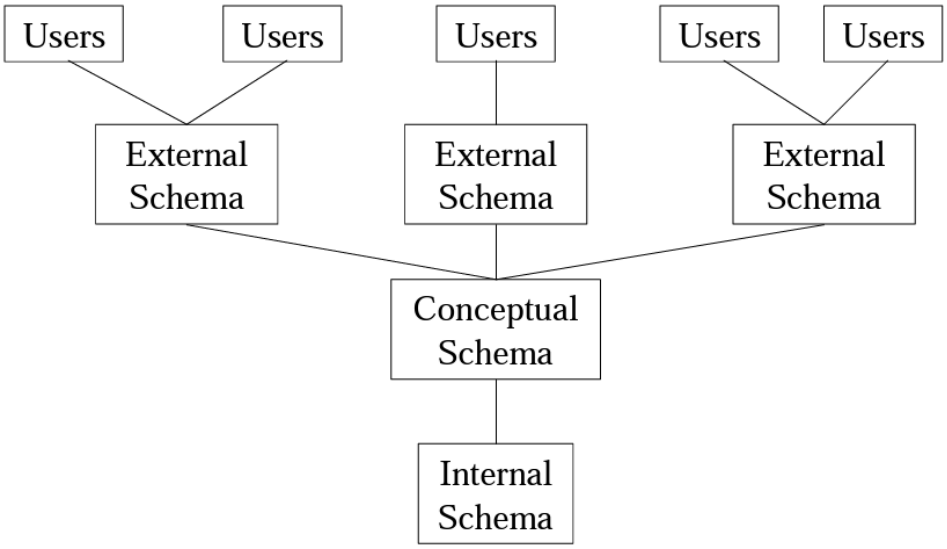
Arquitectura de Bases de Datos Distribuidas

Un sistema de bases de datos distribuida permite que las aplicaciones accedan a datos de bases de datos locales y remotas.

La arquitectura de un sistema define su estructura, generalmente, las arquitecturas DBMS distribuidas se desarrollan en función de tres parámetros:

- Distribución: establece la distribución física de los datos en diferentes sitios.
- Autonomía: indica la distribución del control del sistema de base de datos y el grado en que cada DBMS constituyente puede operar de forma independiente.
- Heterogeneidad: se refiere a la uniformidad o disimilitud de los modelos de datos, los componentes del sistema y las bases de datos.

Arquitectura ANSI/SPARC



ANSI-SPARC(American National Standard Institute - Standards Planning and Requirements Comite).

Divide a un sistema en 3 niveles: interno, conceptual y externo. Los cuales separan los programas de aplicación de la base de datos física.

Interno: Describe la estructura física mediante un modelo físico además de los detalles para la base de datos, así como los métodos de acceso. Es la forma de cómo se almacenan físicamente los datos ya sea por registros o por cualquier otra forma.

Conceptual: Describe la estructura de toda la base de datos para el usuario, describiendo entidades, atributos, relaciones, operaciones de los usuarios y restricciones. Todo desde el punto de vista del mundo real

Se trata de un modelo relacional en los cuales los objetos visibles serán tablas relacionales.

Externo: Describe la parte de base de datos que interesa a un grupo de usuarios determinado y oculta al mismo el resto de la base de datos. Es el cómo percibe el usuario los datos.

Frecuentemente se maneja que será relacional.

David Bell mencionó que se debe definir un modelo de referencia basado en:

- Componentes: Número de componentes y relaciones, representando alguna funcionalidad.
- Funciones. Diferentes clases de usuarios junto con la funcionalidad que el sistema ofrecerá para cada clase, se tiene una estructura jerárquica de clases.
- Datos. Diferentes tipos de descripción de datos, se describen las unidades funcionales que realizarán y/o usarán los datos de acuerdo con las diferentes vistas.

Arquitectura genérica centralizada DBMS

DBMS (Database Management System) es un sistema de gestión de bases de datos ya que una base de datos requiere un programa de software de bases de datos completo. Sirve como interfaz entre la base de datos y los usuarios finales lo que permite a los usuarios recuperar, actualizar y gestionar cómo se organiza y se optimiza la información.

Ejemplos de DBMS MySQL, Microsoft Access, Microsoft SQL Server, FileMaker Pro, Oracle Database y dBASE.

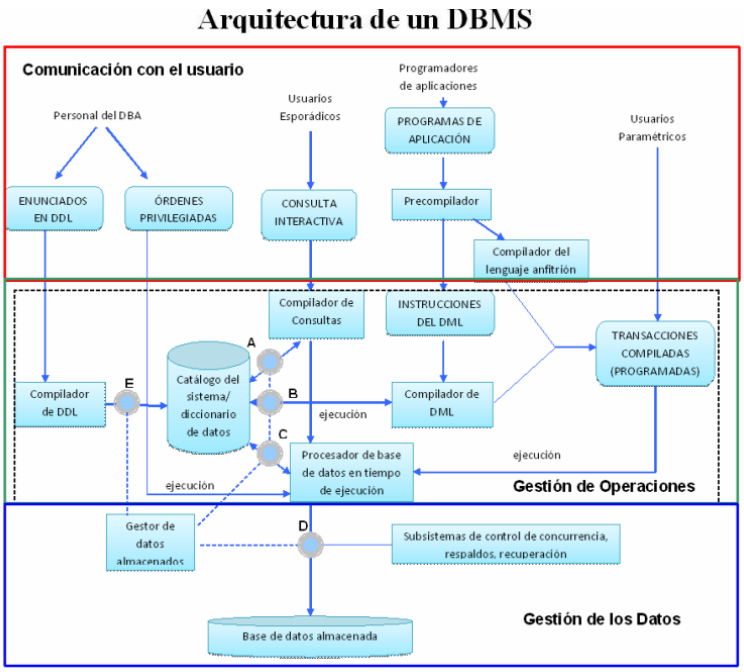
Lo que sucede en un DBMS es lo siguiente:

1. Un usuario emite una solicitud de acceso, usando algún sublenguaje de datos en particular
2. El DBMS acepta esa solicitud y la analiza
3. El DBMS inspecciona el esquema externo para ese usuario el correspondiente mapeo externo/conceptual y la definición de la base de datos almacenada.
4. El DBMS ejecuta la base de datos almacenada

Los objetivos de los DBMS son:

- Abstracción de información
- Independencia
- Consistencia
- Seguridad
- Manejo de transacciones
- Tiempo de respuesta

Las bases de datos respetan la arquitectura de 3 niveles definida, por el grupo ANSI/SPARC.

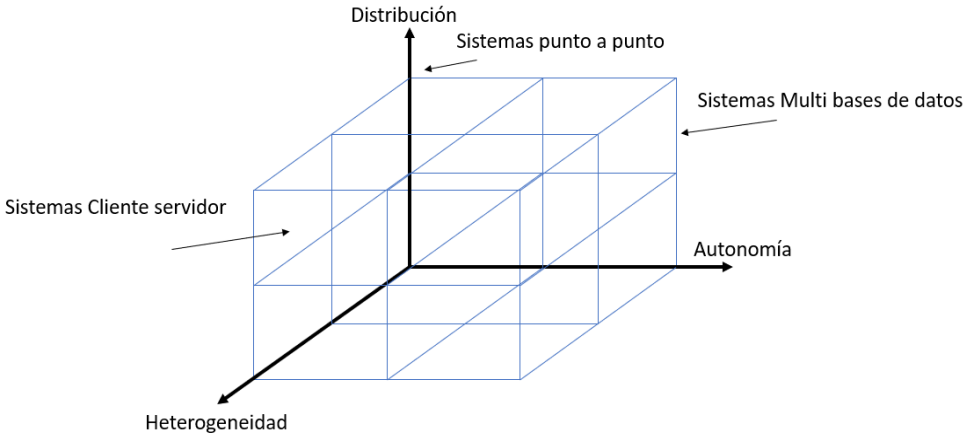


El tipo de arquitectura integrada es consecuencia de los Lenguajes de Definición de datos (DDL) y los de manipulación de datos (DML) en un solo lenguaje, el cual resulta muy cómodo para el DBA puesto que le basta con aprender un solo lenguaje formal para realizar todas las tareas de creación y mantenimiento de la base de datos.

Modelos de arquitectura para DBMSs distribuidas

Se le denomina por las formas de trabajar cooperativamente, donde el sistema se ve caracterizado por:

- Autonomía
  - ❖ Distribución del control
  - ❖ Grado de operación independiente
  - ❖ Requerimientos
    - ➢ Operaciones locales no afectadas por la participación
    - ➢ Como el DBMS procesa queries y las optimiza no debe ser afectado por queries globales
    - ➢ Consistencia y operación no se compromete por el join o leave de DBMS
- Distribución.
  - ❖ Distribución de datos en forma física
  - ❖ Formas
    - ➢ Cliente-Servidor
    - ➢ Peer to Peer
- Heterogeneidad.
  - ❖ Modelos de datos
  - ❖ Lenguajes de Queries
  - ❖ Protocolos de transacciones



Autonomía

Se refiere a la distribución del control, se indica el grado en que los DBMS individuales pueden operar de forma independiente y es una función de una serie de factores, como si los DBMS individuales intercambian información, si pueden ejecutar transacciones de forma independiente y si se permite modificarlas.

Requisitos de un sistema autónomo:

- Las operaciones locales de los DBMS individuales no se ven afectadas por su participación en el sistema distribuido.
- La forma en que los DBMS individuales procesan las consultas y las optimizan *no debe verse afectada* por la ejecución de consultas globales que acceden a múltiples bases de datos.
- La operación del sistema no debe verse comprometida cuando los DBMS individuales se unen o abandonan el sistema distribuido.

Dimensiones de la autonomía

Autonomía de diseño

La capacidad para decidir aspectos relacionados con el diseño así como que DBMS utilizar. Algunos puntos que se toman en consideración son:

- Restricciones utilizadas para gestionar los datos
- Funcionalidades del sistema
- Modelo de datos y lenguaje de consultas

Autonomía de comunicación

Cada DBMS individual es libre de tomar decisiones sobre qué tipo de información quiere proporcionar al otro DBMS o al software que controla su ejecución general.

Autonomía de ejecución

Cada DBMS decide cuándo puede compartir sus recursos con otros componentes, inclusive la capacidad de asociarse o retirarse con otros.

Distribución

La distribución se ocupa de los datos, se considera la distribución física de datos en múltiples sitios, pero el usuario ve estos datos como una única base de datos lógica.

Hay varias formas en que se han distribuido los DBMS; Cliente/Servidor y Peer to Peer:

Cliente/Servidor

Las funciones de gestión de datos se realiza por los servidores, mientras que los clientes se centran en proporcionar el entorno de la aplicación , las tareas de comunicación se comparten entre las máquinas cliente y los servidores.

Hay varias formas de estructurar, cada una proporciona un nivel diferente de distribución.

Peer to Peer

No hay distinción entre máquinas clientes y servidores, cada máquina tiene una funcionalidad DBMS completa y se puede comunicar con otras máquinas para ejecutar consultas

Heterogeneidad

La heterogeneidad se refiere a la uniformidad o disimilitud de los modelos de datos, los componentes del sistema y las bases de datos, se puede convertir en una tarea difícil de lograr, en específico cuando se quiere hacer que trabajen de manera distribuida

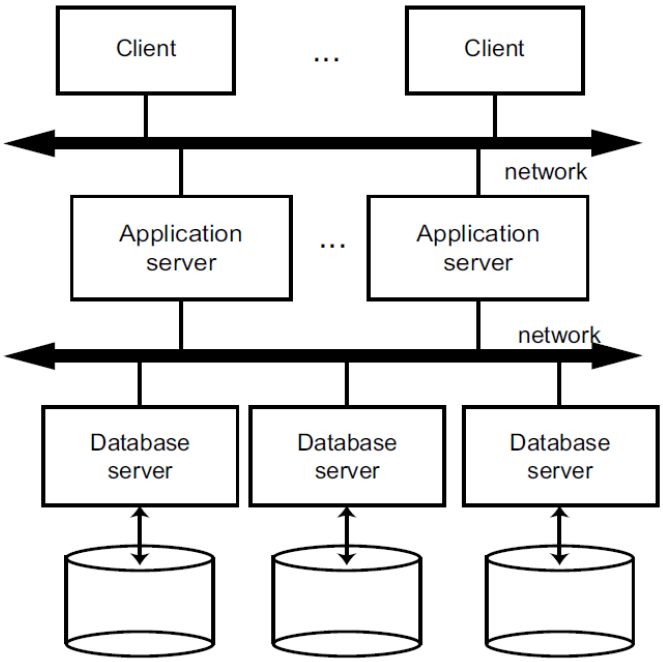
La heterogeneidad puede ocurrir de diversas formas en los sistemas distribuidos, desde la heterogeneidad del hardware y las diferencias en los protocolos de red hasta las variaciones en los administradores de datos.

Para que un sistema administrador de base de datos distribuido pueda llamarse heterogéneo; debe utilizar al menos dos sistemas gestores de bases de datos distintos; estos a su vez presentan un problema en la transferencia de los datos e información ya que cada DBMS tiene su propio modelo de datos.

Sistemas Cliente/Servidor

Un servidor de base de datos es el software que administra una base de datos y un cliente es una aplicación que solicita información de un servidor, cada computadora en una red es un nodo que puede tener una o más bases de datos y cada uno de estos nodos del sistema de bases de datos distribuidas puede actuar como cliente o servidor o como ambos, depende de la situación.

Los datos completos se pueden ver como una única base de datos lógica, mientras que a nivel físico los datos se pueden ser distribuidos.



Cliente: Un Cliente es una computadora o una aplicación de usuario que solicita servicios del servidor, también se conoce como aplicación front-end, ya que el usuario final interactúa con el proceso cliente.

Las propiedades de un cliente son:

- Activo (Maestro)
- Envío de solicitudes
- Espera hasta que llegue la respuesta.

Servidor: Es una o más computadoras o una aplicación de computadora que proporciona servicios a los clientes, también se conoce como aplicación de back-end, ya que el proceso del servidor proporciona los servicios al cliente.

Las propiedades del servidor son:

- Passivo (esclavo)
- Espera solicitudes
- Cuando se hacen solicitudes por clientes envía respuestas

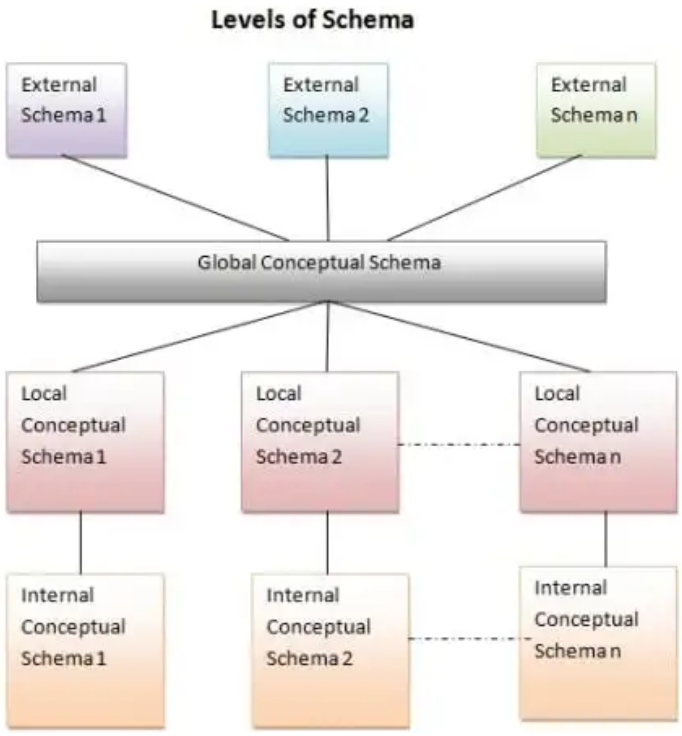
La principal ventaja es que mejora la usabilidad, la flexibilidad, la interoperabilidad y la escalabilidad, tiene otras ventajas como que este tipo de sistemas tienen la habilidad de distribuir la carga de trabajo entre las estaciones de trabajo del cliente y los servidores compartidos.

Sistemas Punto a Punto

En estos sistemas, cada nodo actúa como cliente y como servidor para impartir los servicios de base de datos, por lo tanto cada nodo puede acceder a servicios de otros nodos como también proveer servicios para otros nodos. En diferencia con la arquitectura cliente/servidor, en un sistema distribuido peer-to-peer cada nodo proporciona facilidades de interacción con el usuario así como capacidades de procesamiento.

Teniendo en cuenta la complejidad asociada con la comunicación y la gestión de una gran cantidad de computadoras involucradas en un sistema distribuido, el módulo de software en cada nodo en un sistema distribuido peer-to-peer generalmente está estructurado en cuatro niveles.

- Esquema conceptual global: representa la vista lógica global de los datos.
- Esquema conceptual local: representa la organización lógica de los datos en cada sitio.
- Esquema interno local: representa la organización de datos físicos en cada sitio.
- Esquema externo: representa la vista de los datos del usuario.



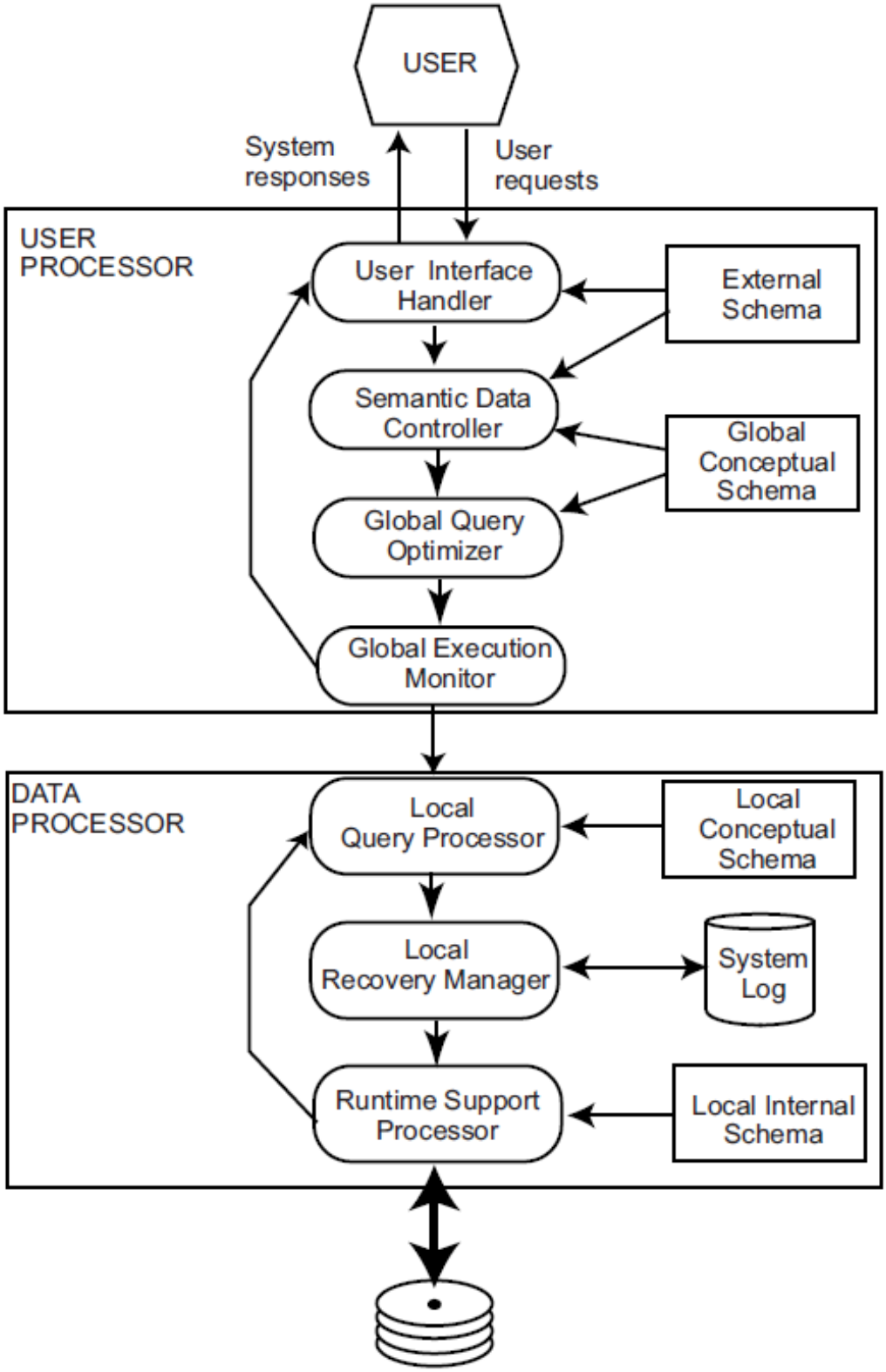
El componente componente principal que maneja la interacción con los usuarios, se conoce como *procesador de usuario*:

- Controlador de la interfaz de usuario: interpreta los comandos del usuario cuando ingresan y formatea los datos de resultados a cuando se envían al usuario.
- Controlador de datos semánticos: utiliza las restricciones que se definen como parte del esquema conceptual global para verificar si la consulta puede procesarse
- Optimizador y descomponedor de consultas globales: trata de generar la mejor estrategia para ejecutar operaciones de unión distribuida buscando reducir costo
- Monitor de ejecución distribuida: coordina la ejecución distribuida de la solicitud del usuario

El componente que se ocupa del almacenamiento, se conoce como *procesador de datos:*

- Optimizador de consultas local: es responsable de elegir la mejor ruta de acceso para cualquier elemento de datos
- Administrador de recuperación local: es responsable de que la base de datos local permanezca consistente incluso cuando ocurran fallas
- Procesador de soporte en tiempo de ejecución: accede físicamente a la base de datos de acuerdo con los comandos físicos en el programa generado por el optimizador de consultas.

En los sistemas peer-to-peer, se espera encontrar tanto los módulos del procesador de usuario como los módulos del procesador de datos en cada máquina.


USER
System responses
User requests
USER PROCESSOR
User Interface Handler
External Schema
Semantic Data Controller
Global Conceptual Schema
Global Query Optimizer
Global Execution Monitor
DATA PROCESSOR
Local Query Processor
Local Conceptual Schema
Local Recovery Manager
System Log
Runtime Support Processor
Local Internal Schema

Conclusiones

Estos sistemas tienen como objetivos el uso correcto de los recursos disponibles tales como el almacenamiento y la disponibilidad, además de distribuir lo mejor posible la carga de trabajo y obtener un mejor procesamiento local.

Además de que al tratarse de un sistema distribuido permite compartir estos mismos recursos, estas arquitecturas al estar basadas en protocolos estándar, nos permite amplia flexibilidad al uso de distintos hardware y software provenientes de distintos vendedores.

Gracias a esto se pueden tener beneficios desde a nivel de usuario para clasificar información que le interese. Podemos manejar todo el sistema DBMS de tal manera que distribuimos las funciones sobre su autonomía, distribución y heterogeneidad.

Referencias

[1] "Distributed DBMS - Features, Needs and Architecture". [online]. Disponible en https://www.csitweb.com/distributed-dbms-features-needs-and-architecture

[2] R. Chhanda. *Distributed Database Systems*. Pearson Education India, 2008.

[3] Özsu, M. Tamer, Patrick Valduriez. *Principles of Distributed Database Systems*. Pearson Education, 2011.

[4] E. G. Flores Castro, "Implementación de una base de datos heterogénea distribuida entre los SGBDs ORACLE, MySQL y PostgreSQL con replicación, mediante un script bash implementado en el sistema operativo CentOS usando software libre.", INNOVA Research Journal, vol. 3, pp. 68–77, febrero de 2018.

[5] Ermeneses,Bases de datos Avanzadas, Facultad de estadística e Informática, pp 5-19,2017 [online] Available: https://www.uv.mx/personal

[6] A. Reyes, Aruitecturas ANSI/SPARK, pp. 1-5 [online] Disponible en: http://aisii.azc.uam.mx

[7] Oracle(2022) Temas de base de datos, Disponible en: https://www.oracle.com/mx

[8] Y. Camargo y F. Lugo(2013,Mayo 26) Características, componentes y arquitectura de los dbms [online] Disponible en: https://es.slideshare.net

[9] Y. Breitbart, H. Korth, A. Silberschatz y S. Sudarshan. Distributed Databases, en Encyclopedia of Electrical and Electronics Engineering. John Wiley and Sons(1999) .

[10] Hugo D(2006,Septiembre 19) Base de datos distribuidas[online] Disponible en: http://weblidi.info.unlp.edu.ar/