**ESQUIMAS RIGIDOS EN BASES DE DATOS Y ESQUEMAS FLEXIBLES EN BASES DE DATOS NO SQL**

ANGEL ESTEBAN VILLAMIL ROJAS

UNIVERSIDAD UNINPAHU

INGENIERÍA EN SOFTWARE

PROFESORA: LUIS REYES

ABRIL DE 2025

**ESQUEMAS RÍGIDOS EN BASES DE DATOS SQL VS. ESQUEMAS FLEXIBLES EN NOSQL**

Las bases de datos tradicionales SQL utilizan esquemas rígidos que requieren una estructura predefinida, mientras que las bases de datos NoSQL permiten esquemas flexibles, adaptándose a modelos de datos dinámicos. Este documento compara ambos enfoques, destacando sus características, ventajas y desventajas, con referencias según normas APA.

Esquemas Rígidos en Bases de Datos SQL

Características

Estructura predefinida: Las tablas, columnas, tipos de datos y relaciones deben definirse antes de insertar datos (Date, 2004).

Validación estricta: Los datos deben cumplir con los tipos definidos (e.g., INT, VARCHAR, DATE) (Elmasri & Navathe, 2015).

Relaciones explícitas: Se usan claves primarias (PRIMARY KEY) y foráneas (FOREIGN KEY) para mantener la integridad referencial (Silberschatz et al., 2010).

Ventajas

✅ Integridad de datos: Garantiza consistencia mediante propiedades ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) (Haerder & Reuter, 1983).

✅ Optimización de consultas: Índices y joins mejoran el rendimiento en consultas complejas (Garcia-Molina et al., 2008).

✅ Documentación implícita: El esquema sirve como documentación de la estructura de datos (Coronel & Morris, 2018).

Desventajas

❌ Poca flexibilidad: Modificar esquemas (añadir/eliminar columnas) requiere migraciones costosas (Stonebraker & Cetintemel, 2005).

❌ Dificultad con datos semiestructurados: No maneja bien JSON, XML o datos anidados sin normalización (Cattell, 2011).

Esquemas Flexibles en Bases de Datos NoSQL

Características

Sin esquema fijo: Los datos pueden variar en estructura (MongoDB permite documentos JSON diferentes en una misma colección) (Chodorow, 2013).

Tipos de bases NoSQL:

Documentales (MongoDB, CouchDB): Almacenan registros en documentos flexibles (Banker et al., 2013).

Clave-Valor (Redis, DynamoDB): Sin esquema, solo pares clave-valor (Vukotic et al., 2015).

Grafos (Neo4j): Enfocados en relaciones más que en estructura fija (Robinson et al., 2015).

Columnas anchas (Cassandra, HBase): Permiten diferentes columnas por fila (Hewitt, 2010).

Ventajas

✅ Escalabilidad horizontal: Distribución en clusters para alto rendimiento (Abadi, 2012).

✅ Adaptabilidad: Ideal para datos semiestructurados o cambiantes (Sadalage & Fowler, 2012).

✅ Rendimiento en escritura: Optimizado para alta velocidad de inserción (Grolinger et al., 2013).

Desventajas

❌ Consistencia eventual (BASE): No garantiza ACID, puede haber inconsistencia temporal (Brewer, 2012).

❌ Dificultad en consultas complejas: Sin joins nativos, requiere denormalización (McCormick et al., 2015).

Referencias (Normas APA)

* Abadi, D. (2012). NoSQL databases: Why, what, and when. O’Reilly Media.
* Brewer, E. (2012). CAP twelve years later: How the "rules" have changed. Computer, 45(2), 23-29.
* Cattell, R. (2011). Scalable SQL and NoSQL data stores. ACM SIGMOD Record, 39(4), 12-27.
* Chodorow, K. (2013). MongoDB: The Definitive Guide. O’Reilly.
* Coronel, C., & Morris, S. (2018). Database Systems: Design, Implementation, & Management. Cengage.
* Date, C. J. (2004). An Introduction to Database Systems. Pearson.
* Elmasri, R., & Navathe, S. B. (2015). Fundamentals of Database Systems. Pearson.
* Grolinger, K., et al. (2013). NoSQL databases for big data. IEEE International Conference on Big Data.
* Hewitt, E. (2010). Cassandra: The Definitive Guide. O’Reilly.
* Sadalage, P., & Fowler, M. (2012). NoSQL Distilled. Addison-Wesley.