Sistemas SQL (finales de década de 1970

IBM comenzó a trabajar a principios de 1970 en un prototipo lejanamente basado en los conceptos de Codd llamándolo System R. La primera versión estuvo lista en 1974 o 1975, y comenzó así el trabajo en sistemas multi-tabla, en los que los datos podían disgregarse de modo que toda la información de un registro (alguna de la cual es opcional) no tiene que estar almacenada en un único trozo grande. Las versiones multi-usuario siguientes fueron probadas por los usuarios en 1978 y 1979, tiempo por el que un lenguaje SQL había sido estandarizado. Las ideas de Codd se revelaron como operativas y superiores a las de CODASYL, lanzando a IBM al desarrollo de una verdadera versión de producción de System R, conocido como SQL/DS, y posteriormente como Database 2 (DB2).

Muchos de los técnicos de INGRES estaban seguros del éxito comercial del sistema, y formaron sus propias compañías para comercializar el desarrollo pero con una interfaz SQL. Sybase, Informix , NonStop SQL y la misma INGRES se vendían como derivados del INGRES original en los años 1980. Incluso el SQL Server de Microsoft está basado en Sybase, y por consiguiente en INGRES. Sólo Larry Ellison -el fundador de Oracle- comenzó un nuevo camino basado en el artículo de IBM sobre System R, y aventajó a IBM sacando al mercado su primera versión en 1978.

Stonebraker aplicó las lecciones de INGRES al desarrollo de una nueva base de datos -Postgres- conocida ahora como PostgreSQL. PostgreSQL se utiliza para muchas aplicaciones críticas (los registros de dominios .org y .info lo usan para su almacenamiento primario, así como grandes compañías e instituciones financieras).

En Suecia, el artículo de Codd generó la base de datos Mimer SQL[3] en la universidad de Uppsala. En 1984 este proyecto se consolidó en una compañía independiente. A principios de 1980, Mimer introdujo la gestión de transacciones para dar robustez a las aplicaciones, una idea que fue recogida en muchos otros SGBD.

Sistemas orientados a objetos (1980)

Durante la década de 1980 el auge de la programación orientada a objetos influyó en el modo de manejar la información de las bases de datos. Programadores y diseñadores comenzaron a tratar los datos en las bases de datos como objetos. Esto quiere decir que si los datos de una persona están en la base de datos, los atributos de la persona como dirección, teléfono y edad se consideran que pertenecen a la persona, no son datos extraños. Esto permite establecer relaciones entre objetos y atributos, más que entre campos individuales.

Otro gran foco de atención durante la década fue el incremento de velocidad y fiabilidad en el acceso. En 1989, dos profesores de la Universidad de Wisconsin publicaron un artículo en una conferencia ACM en el que exponían sus métodos para mejorar las prestaciones de las bases de datos. La idea consistía en replicar la información importante -y más solicitada- en una base de datos temporal de pequeño tamaño con enlaces a la base de datos principal. Esto implicaba que se podía buscar mucho más rápido en la base de datos pequeña que en la grande. Su mejora de prestaciones llevó a la introducción de la indización, incorporado en la totalidad de los SGBD.

Sistemas NoSQL (2000)

El siglo XXI trajo una nueva tendencia en las bases de datos: el NoSQL. Esta tendencia introducía una línea no relacional significativamente diferentes de las clásicas. No requieren por lo general esquemas fijos, evitan las operaciones *join* almacenando datos desnormalizados y están diseñadas

para escalar horizontalmente. La mayor parte de ellas pueden clasificarse como almacenes clave-valor o bases de datos orientadas a documentos.

Recientemente ha habido una gran demanda de bases de datos distribuidas con tolerancia a particiones, pero de acuerdo con el teorema CAP no es posible conseguir un sistema distribuido que simultáneamente proporcione consistencia, disponibilidad y tolerancia al particionado. Un sistema distribuido puede satisfacer sólo dos de las tres restricciones a la vez. Por dicha razón muchas de las bases de datos NoSQL usan la llamada consistencia eventual para proporcionar disponibilidad y tolerancia al particionado, con un nivel máximo de consistencia de datos.

Entre las aplicaciones más populares encontramos MongoDB, MemcacheDB, Redis, CouchDB, Hazelcast, Apache Cassandra y HBase, todas ellas de código abierto.