Bases de Datos I
22/09/2020
Resumen Cap 3 22/09
IS-501
José Inestroza

Capítulo 01 SQL Complete Reference

1- El lenguaje SQL

- Definición de datos SQL permite al usuario definir la estructura y organización de los datos almacenados y las relaciones entre los elementos de datos almacenados.
- Recuperación de datos SQL permite a un usuario o un programa de aplicación recuperar datos almacenados de la base de datos y utilizarlos.
- La manipulación de datos SQL permite que un usuario o un programa de aplicación actualice la base de datos agregando datos nuevos, eliminando datos antiguos y modificando datos almacenados previamente.
- El SQL de control de acceso se puede utilizar para restringir la capacidad de un usuario de recuperar, agregar y modificar datos, protegiendo los datos almacenados contra el acceso no autorizado.
- SQL para compartir datos se utiliza para coordinar el intercambio de datos entre usuarios simultáneos, lo que garantiza que los cambios realizados por un usuario no borren inadvertidamente los cambios realizados casi al mismo tiempo por otro usuario.
- Integridad de datos SQL define las restricciones de integridad en la base de datos, protegiéndola de la corrupción debido a actualizaciones inconsistentes o fallas del sistema.

El Rol de SQL

- SQL es un lenguaje de consulta interactivo. Los usuarios escriben comandos SQL en un programa SQL interactivo para recuperar datos y mostrarlos en la pantalla, proporcionando una herramienta conveniente y fácil de usar para consultas de bases de datos ad hoc.
- SQL es un lenguaje de programación de bases de datos. Los programadores incorporan comandos SQL en sus programas de aplicación para acceder a los datos en una base de datos. Tanto los programas escritos por el usuario como los programas de utilidad de base de datos (como los redactores de informes y las herramientas de entrada de datos) utilizan esta técnica para acceder a la base de datos.
- SQL es un lenguaje de administración de bases de datos. El administrador de la base de datos responsable de administrar una base de datos de miniordenador o mainframe utiliza SQL para definir la estructura de la base de datos y controlar el acceso a los datos almacenados.
- SQL es un lenguaje cliente / servidor. Los programas de computadora personal usan SQL para comunicarse a través de una red con servidores de bases de datos que almacenan datos compartidos. Esta arquitectura cliente / servidor es utilizada por muchas aplicaciones populares de clase empresarial.
- SQL es un lenguaje de acceso a datos de Internet. Los servidores web de Internet que interactúan con los datos corporativos y los servidores de aplicaciones de Internet utilizan SQL como lenguaje estándar para acceder a las bases de datos corporativas, a menudo incorporando el acceso a la base de datos SQL en lenguajes de scripting populares como PHP o Perl.

- SQL es un lenguaje de base de datos distribuido. Los sistemas de administración de bases de datos distribuidas utilizan SQL para ayudar a distribuir datos en muchos sistemas informáticos conectados. El software DBMS en cada sistema usa SQL para comunicarse con los otros sistemas, enviando solicitudes de acceso a datos.
- SQL es un lenguaje de puerta de enlace de base de datos. En una red informática con una combinación de diferentes productos DBMS, SQL se utiliza a menudo en una puerta de enlace que permite que una marca de DBMS se comunique con otra marca.

1- Fuentes clave SQL

Éstos son algunas de las características principales y fuerzas del mercado que han contribuido a este éxito durante los últimos 25 años:
• Independencia del proveedor
• Portabilidad entre sistemas informáticos
• Estándares oficiales de SQL
• Compromiso temprano de IBM
• Soporte de Microsoft
• Base relacional
• Estructura de alto nivel similar al inglés
• Consultas interactivas y ad hoc
• Acceso a la base de datos programática
• Varias vistas de datos
• Idioma completo de la base de datos
• Definición de datos dinámicos
• Arquitectura cliente / servidor
• Soporte de aplicaciones empresariales
• Extensibilidad y tecnología de objetos
• Acceso a la base de datos de Internet
• Integración de Java (JDBC)
• Soporte de código abierto
• Infraestructura industrial

Recuperando datos
La instrucción SQL que recupera datos de la base de datos se llama SELECT.
Esta declaración SQL recupera los datos que desea:

➔ **SELECT City, Office, Sales FROM OFFICES;**

La instrucción SELECT se utiliza para todas las consultas SQL. Por ejemplo, aquí hay una consulta que enumera los nombres y las ventas hasta la fecha de cada vendedor en la base de datos. También muestra la cuota (objetivo de ventas) y el número de oficina donde trabaja cada persona. En este caso, los datos provienen de la tabla SALESREPS

➔ **SELECT NAME, REP_OFFICE, SALES, QUOTA FROM SALESREPS;**

Es nuevo en la empresa y aún no ha sido asignado a una oficina de ventas ni se le ha asignado una cuota de ventas. Sin embargo, ya ha realizado algunas ventas. Los datos en su fila de resultados de la consulta lo muestran claramente.
SQL también le permite solicitar resultados calculados. Por ejemplo, puede pedirle a SQL que calcule la cantidad por la cual cada vendedor está por encima o por debajo de la cuota:

➔ **SELECT NAME, SALES, QUOTA, (SALES – QUOTA) FROM SALESREPS;**

Quizás le gustaría centrarse en los vendedores cuyas ventas son inferiores a sus cuotas. SQL le permite recuperar ese tipo de información selectiva muy fácilmente, agregando una comparación matemática a la solicitud anterior:

➔ **SELECT NAME, SALES, QUOTA, (SALES – QUOTA) FROM SALESREPS WHERE SALES < QUOTA;**

Puede usar la misma técnica para enumerar pedidos grandes en la base de datos y averiguar qué cliente realizó el pedido, qué producto se pidió y en qué cantidad. También puede pedirle a SQL que ordene los pedidos según el monto del pedido:

➔ **SELECT ORDER_NUM, CUST, PRODUCT, QTY, AMOUNT FROM ORDERS WHERE AMOUNT > 25000.00 ORDER BY AMOUNT;**

ORDERS Table

ORDER_NUM	CUST	PRODUCT	QTY	AMOUNT
112961	2117	2A44L	7	$31,500.00
113012	2111	41003	35	$3,745.00
112989	2101	114	6	$1,458.00
113051	2118	XK47	4	$1,420.00
112968	2102	41004	34	$3,978.00
113036	2107	4100Z	9	$22,500.00
113045	2112	2A44R	10	$45,000.00
112963	2103	41004	28	$3,276.00
113013	2118	41003	1	$652.00
113058	2108	112	10	$1,480.00
112997	2124	41003	1	$652.00
112983	2103	41004	6	$702.00
113024	2114	XK47	20	$7,100.00
113062	2124	114	10	$2,430.00
112979	2114	4100Z	6	$15,000.00
113027	2103	41002	54	$4,104.00
113007	2112	773C	3	$2,925.00
113069	2109	775C	22	$31,350.00
113034	2107	2A45C	8	$632.00
112992	2118	41002	10	$760.00
112975	2111	2A44G	6	$2,100.00
113055	2108	4100X	6	$150.00
113048	2120	779C	2	$3,750.00
112993	2106	2A45C	24	$1,896.00
113065	2106	XK47	6	$2,130.00
113003	2108	779C	3	$5,625.00
113049	2118	XK47	2	$776.00
112987	2103	4100Y	11	$27,500.00
113057	2111	4100X	24	$600.00
113042	2113	2A44R	5	$22,500.00

CUSTOMERS Table

CUST_NUM	COMPANY	CUST_REP	CREDIT_LIMIT
2111	JCP Inc.	103	$50,000.00
2102	First Corp.	101	$65,000.00
2103	Acme Mfg.	105	$50,000.00
2123	Carter & Sons	102	$40,000.00
2107	Ace International	110	$35,000.00
2115	Smithson Corp.	101	$20,000.00
2101	Jones Mfg.	106	$65,000.00
2112	Zetacorp	108	$50,000.00
2121	QMA Assoc.	103	$45,000.00
2114	Orion Corp.	102	$20,000.00
2124	Peter Brothers	107	$40,000.00
2108	Holm & Landis	109	$55,000.00
2117	J.P. Sinclair	106	$35,000.00
2122	Three-Way Lines	105	$30,000.00
2120	Rico Enterprises	102	$50,000.00
2106	Fred Lewis Corp.	102	$65,000.00
2119	Solomon Inc.	109	$25,000.00
2118	Midwest Systems	108	$60,000.00
2113	Ian & Schmidt	104	$20,000.00
2109	Chen Associates	107	$25,000.00
2105	AAA Investments	101	$45,000.00

SALESREPS Table

NAME	REP_OFFICE	QUOTA	SALES
Bill Adams	13	$350,000.00	$367,911.00
Mary Jones	11	$300,000.00	$392,725.00
Sue Smith	21	$350,000.00	$474,050.00
Sam Clark	11	$275,000.00	$299,912.00
Bob Smith	12	$200,000.00	$142,594.00
Dan Roberts	12	$300,000.00	$305,673.00
Tom Snyder	NULL	NULL	$75,985.00
Larry Fitch	21	$350,000.00	$361,865.00
Paul Cruz	12	$275,000.00	$286,775.00
Nancy Angelli	22	$300,000.00	$186,042.00

OFFICES Table

OFFICE	CITY	REGION	TARGET	SALES
22	Denver	Western	$300,000.00	$186,042.00
11	New York	Eastern	$575,000.00	$692,637.00
12	Chicago	Eastern	$800,000.00	$735,042.00
13	Atlanta	Eastern	$350,000.00	$367,911.00
21	Los Angeles	Western	$725,000.00	$835,915.00

Resumen de datos

SQL no solo recupera datos individuales de la base de datos, sino que también puede resumir el contenido de la base de datos. ¿Cuál es el tamaño medio de un pedido en la base de datos? Esta solicitud le pide a SQL que mire todos los pedidos y encuentre el monto promedio:

➔ **SELECT AVG(AMOUNT) FROM ORDERS;**

Finalmente, averigüemos el valor total de los pedidos realizados por cada cliente. Para hacer esto, puede pedirle a SQL que agrupe los pedidos por número de cliente y luego totalice los pedidos de cada cliente:

- ➔ **SELECT CUST, SUM(AMOUNT) FROM ORDERS GROUP BY CUST;**

Agregar datos a la base de datos
También usa SQL para agregar nuevos datos a la base de datos. Por ejemplo, suponga que acaba de abrir una nueva oficina de ventas de la región occidental en Dallas, con ventas objetivo de 275.000 dólares. Aquí está la declaración INSERT que agrega la nueva oficina a la base de datos, como oficina número 23:

- ➔ **INSERT INTO OFFICES (CITY, REGION, TARGET, SALES, OFFICE) VALUES ('Dallas', 'Western', 275000.00, 0.00, 23);**

- ➔ **INSERT INTO CUSTOMERS (COMPANY, CUST_REP, CUST_NUM, CREDIT_LIMIT) VALUES ('Acme Industries', 109, 2125, 25000.00);**

Eliminar datos
Así como la instrucción SQL INSERT agrega nuevos datos a la base de datos, la instrucción SQL DELETE elimina datos de la base de datos. Si Acme Industries decide unos días después cambiar a un competidor, puede eliminar la información del cliente de Acme de la base de datos con esta declaración:

- ➔ **DELETE FROM CUSTOMERS WHERE COMPANY = 'Acme Industries';**

- ➔ **DELETE FROM SALESREPS WHERE SALES < QUOTA;**

Actualización de la base de datos
Puede utilizar SQL para modificar datos que ya están almacenados en la base de datos. Por ejemplo, para aumentar el límite de crédito para First Corp. a $ 75,000, usaría la instrucción SQL UPDATE:

- ➔ **UPDATE CUSTOMERS SET CREDIT_LIMIT = 75000.00 WHERE COMPANY = 'First Corp.';**

- ➔ **UPDATE SALESREPS SET QUOTA = QUOTA + 15000.00;**

Protección de datos
Una función importante de una base de datos es proteger los datos almacenados del acceso o modificación por parte de usuarios no autorizados. Por ejemplo, suponga que su asistente, Mary, tiene una cuenta de base de datos pero no ha sido previamente autorizada para insertar datos sobre nuevos clientes en la base de datos. Esta declaración SQL le otorga ese permiso:

- ➔ **GRANT INSERT ON CUSTOMERS TO MARY;**

- ➔ **GRANT UPDATE, SELECT ON CUSTOMERS TO MARY;**

Si a Mary ya no se le permite agregar nuevos clientes a la base de datos, esta declaración REVOKE no lo permitirá:

➔ **REVOKE INSERT ON CUSTOMERS FROM MARY;**

De manera similar, esta declaración REVOKE revocará todos los privilegios de Mary para acceder o modificar los datos del cliente de cualquier manera:

➔ **REVOKE ALL ON CUSTOMERS FROM MARY;**

Creación de base de datos
Antes de poder almacenar datos en una base de datos, primero debe definir la estructura de los datos. Suponga que desea expandir la base de datos de muestra agregando una tabla de datos sobre los productos que vende su empresa. Para cada producto, los datos que se almacenarán incluyen el siguiendo:
• Un código de identificación del fabricante de tres caracteres
• Un código de identificación de producto de cinco caracteres
• Una descripción del producto de hasta 30 caracteres
• El precio del producto
• La cantidad actualmente disponible

➔ CREATE TABLE PRODUCTS
(MFR_ID CHAR(3),
PRODUCT_ID CHAR(5),
DESCRIPTION VARCHAR(30),
PRICE DECIMAL(9,2),
QTY_ON_HAND INTEGER);

Resumen
Este recorrido rápido de SQL le mostró lo que SQL puede hacer e ilustró el estilo de SQL mediante el uso de ocho de las sentencias de SQL más utilizadas. Para resumir:
• Utilice SQL para recuperar datos de la base de datos mediante la instrucción SELECT. Puede recuperar todos o parte de los datos almacenados, ordenarlos y pedirle a SQL que resuma los datos, utilizando totales y promedios.
• Utilice SQL para actualizar la base de datos, agregando nuevos datos con la instrucción INSERT, eliminando datos con la instrucción DELETE y modificando los datos existentes con la instrucción UPDATE.
• Utilice SQL para controlar el acceso a la base de datos, otorgando y revocando privilegios específicos para usuarios específicos con las declaraciones GRANT y REVOKE.
• Utilice SQL para crear / modificar la base de datos definiendo la estructura de nuevas tablas y descartando tablas cuando ya no sean necesarias, utilizando las sentencias CREATE y DROP.

1- SQL en perspectiva

Una breve historia de SQL
En 1974 y 1975, la primera fase del proyecto System / R produjo un prototipo mínimo de un DBMS relacional. Además del DBMS en sí, el proyecto System / R incluyó trabajo en lenguajes de consulta de bases de datos. Uno de estos idiomas se llamó SEQUEL, un acrónimo de Structured English Query Language. En 1976 y 1977, el prototipo de investigación de System / R se reescribió desde cero y la nueva implementación se distribuyó a IBM seleccionado
clientes para evaluación en 1978 y 1979.

Year	Event
1970	Codd defines relational database model
1974	IBM begins System/R project
1974	First article describing the SEQUEL language is published
1978	System/R customer tests are conducted
1979	Oracle introduces first commercial RDBMS
1981	Relational Technology introduces Ingres
1981	IBM announces SQL/DS
1982	ANSI forms SQL standards committee
1983	IBM announces DB2
1986	ANSI SQL1 standard is ratified
1986	Sybase introduces RDBMS for transaction processing
1987	ISO SQL1 standard is ratified
1988	Ashton-Tate and Microsoft announce SQL Server for OS/2
1989	First TPC benchmark (TPC-A) is published
1990	TPC-B benchmark is published
1991	SQL Access Group database access specification is published
1992	Microsoft publishes ODBC specification
1992	ANSI SQL2 standard (SQL-92) is ratified
1992	TPC-C (OLTP) benchmark is published
1993	Specialized SQL data warehousing systems are shipped for the first time
1993	ODBC products are shipped for the first time
1994	Parallel database server technology is shipped commercially
1995	Open source MySQL first released
1996	Standard API for OLAP database access and OLAP benchmark is published
1997	IBM DB2 UDB unifies DB2 architecture across IBM and other vendor platforms
1997	Major DBMS vendors announce Java integration strategies
1998	Microsoft SQL Server 7 provides enterprise-level database support for Windows NT
1998	Oracle 8i provides database/Internet integration and moves away from client/server model
1998	Commercial in-memory database products are shipped for the first time
1999	J2EE standardizes JDBC database access from application servers
1999	ANSI/ISO SQL:1999 standard ratified, adding object-oriented constructs into the language
2000	Oracle introduces application servers with integrated database caching
2000	Microsoft introduces SQL Server 2000, aimed at enterprise applications
2001	XML integration capabilities appear in mainstream RDBMS products
2001	IBM acquires Informix database business
2002	Gartner ranks IBM as #1 database vendor, passing Oracle
2003	ANSI/ISO SQL:2003 ratified, adding SQL/XML
2006	ANSI/ISO SQL:2006 ratified, significantly expanding SQL/XML and object-oriented constructs
2006	IDC and Gartner studies show Oracle leading in market share
2008	MySQL AB acquired by Sun Microsystems
2008	ANSI/ISO SQL:2008 ratified

Productos IBM

Mientras Oracle e Ingres se apresuraban a convertirse en productos comerciales, el proyecto System / R de IBM también se había convertido en un esfuerzo para construir un

producto comercial, llamado SQL / Data System (SQL / DS). IBM anunció SQL / DS en 1981 y comenzó a distribuir el producto en 1982. En 1983, IBM
anunció una versión de SQL / DS para VM / CMS, un sistema operativo que se usaba con frecuencia en los mainframes de IBM en aplicaciones de centros de información corporativos. En 1983, IBM también introdujo Database 2 (DB2), otro DBMS relacional para sus sistemas mainframe. DB2 operaba bajo el sistema operativo MVS de IBM, el sistema operativo de caballo de batalla utilizado en grandes centros de datos de mainframe. El primer lanzamiento de DB2 comenzó a distribuirse en 1985 y los funcionarios de IBM lo elogiaron como una pieza estratégica de la tecnología de software de IBM.

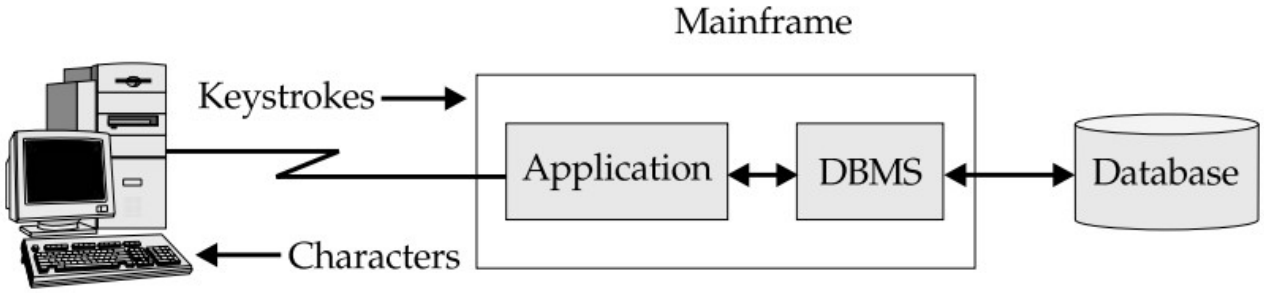
SQL y portabilidad

Ejemplos de áreas donde surgen estas diferencias incluyen

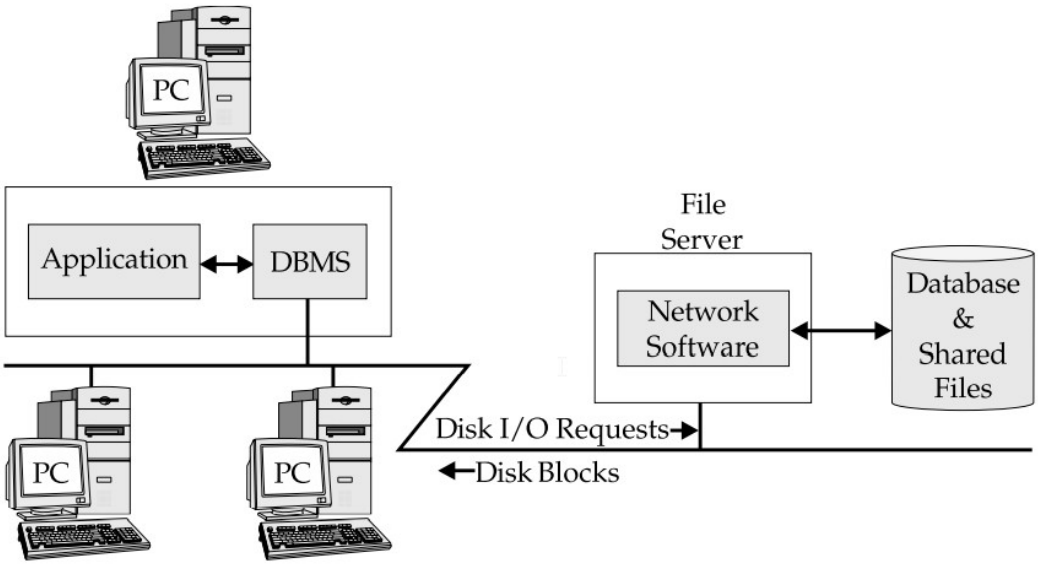
- **Tipos de datos** El estándar SQL ha evolucionado para abordar un conjunto cada vez más amplio de tipos de datos, pero los proveedores siguen agregando nuevos. Incluso los tipos de datos más antiguos pueden causar problemas de portabilidad; el tipo de datos NUMBER de Oracle, por ejemplo, es el más utilizado para representar datos numéricos en una base de datos de Oracle, y sus peculiaridades son completamente exclusivas de Oracle.
- **Compatibilidad con versiones anteriores** No es raro que las aplicaciones empresariales sigan utilizándose 10 o 20 años después de que se escribieron por primera vez, mucho después de que los programadores que las desarrollaron se hayan ido. Estos programas tienden a volverse "intocables", ya que a menudo se ha perdido el conocimiento detallado de cómo funcionan. Grandes secciones de estos programas pueden depender de proveedores de bases de datos y características de SQL patentadas más antiguas se ven obligados a mantener la compatibilidad con versiones anteriores o se arriesgan a "romper" las aplicaciones. Esto perpetúa las diferencias de dialecto que inhiben la portabilidad.
- **Tablas del sistema** El estándar SQL abordó las tablas del sistema que proporcionan información sobre la estructura de la base de datos en sí, comenzando con el estándar SQL-92. En ese momento, los proveedores de bases de datos habían construido sus propias estructuras de tablas de sistemas patentados y han continuado evolucionando, a menudo contienen información útil que va mucho más allá de los elementos especificados en el estándar. Las aplicaciones que utilizan estas tablas del sistema propietario no son portátiles.
- **Interfaz programática** El primer estándar SQL especificaba una técnica abstracta para usar SQL desde un programa de aplicaciones escrito en COBOL, C, FORTRAN y otros lenguajes de programación, que no fue ampliamente adoptado. El estándar SQL / CLI de 1995 finalmente abordó el acceso SQL programático, pero para entonces, los productos DBMS comerciales habían popularizado las interfaces propietarias y las habían integrado profundamente en cientos de miles de aplicaciones de usuario y paquetes de aplicaciones. Aunque las API estándar ahora son ampliamente compatibles, la mayoría de los proveedores de bases de datos aún mantienen interfaces propietarias que ofrecen un mayor rendimiento y una funcionalidad más rica, con el efecto secundario de bloquear las aplicaciones.
- **Diferencias semánticas** Debido a que los estándares especifican ciertos detalles como definidos por el implementador, es posible ejecutar la misma consulta en dos implementaciones de SQL conformes diferentes y producir dos conjuntos diferentes de resultados de consulta. Se pueden encontrar ejemplos de estas diferencias en áreas como el manejo de valores NULL, funciones de columna y eliminación de filas duplicadas.

- **Replicación y duplicación de datos** Muchas bases de datos de producción contienen tablas que se replican en dos o más bases de datos separadas geográficamente, para proporcionar alta disponibilidad o recuperación ante desastres, para distribuir las cargas de trabajo de procesamiento o para reducir los retrasos en la red. Las técnicas para especificar y administrar estos esquemas de replicación son propiedad de cada sistema de base de datos y se han abandonado los intentos de estandarizar la replicación.
- **Códigos de error** El estándar SQL-92 introdujo códigos de error estándar que se devuelven cuando SQL detecta un error, pero todos los sistemas de bases de datos populares llevaban mucho tiempo usando sus propios códigos de error patentados en ese momento. Incluso cuando se utiliza en un modo con códigos de error estándar, las extensiones propietarias pueden generar sus propios errores que están fuera de los códigos estándar especificados.
- **Estructura de la base de datos** El estándar SQL-89 especifica el lenguaje SQL que se utilizará una vez que una base de datos en particular se haya abierto y esté lista para su procesamiento. Los detalles de la nomenclatura de la base de datos y cómo se establece la conexión inicial a la base de datos ya eran diversos y no portátiles cuando se escribió este estándar inicial. El estándar SQL-92 creó más uniformidad, pero el estándar no puede enmascarar completamente estos detalles de implementación.

Arquitectura centralizada
En esta arquitectura, el DBMS y los datos físicos residen en un miniordenador central o en un sistema de mainframe, junto con el programa de aplicación que acepta la entrada desde el terminal del usuario y muestra datos en la pantalla del usuario. El programa de aplicación se comunica con el DBMS mediante SQL.

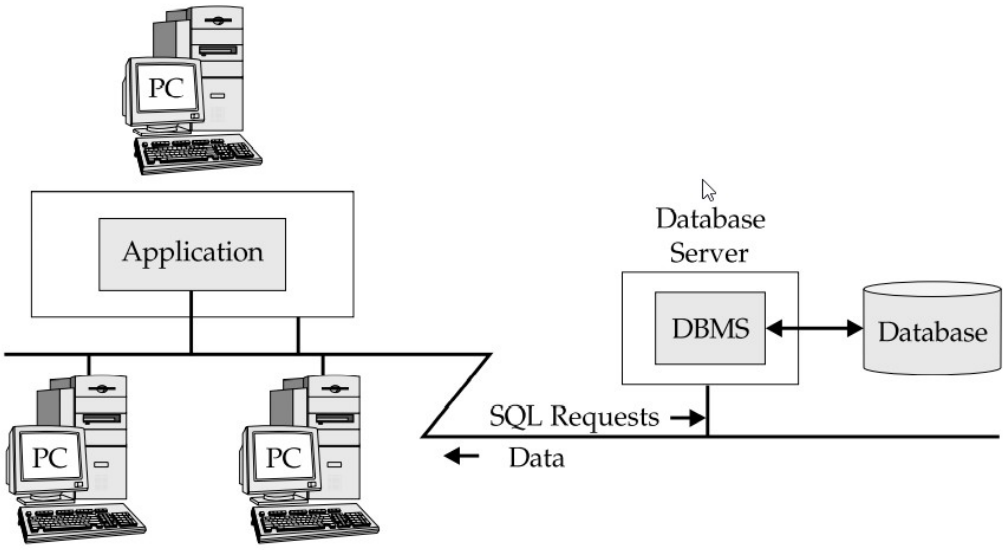


Arquitectura del servidor de archivos
La introducción de computadoras personales y redes de área local condujo al desarrollo de la arquitectura del servidor de archivos, En esta arquitectura, una aplicación que se ejecuta en una computadora personal puede acceder de manera transparente a los datos ubicados en un servidor de archivos, que almacena archivos compartidos. Cuando una aplicación de PC solicita datos de un archivo compartido, el software de red recupera automáticamente el bloque solicitado del archivo del servidor. Las primeras bases de datos de PC, como dBASE y más tarde Access de Microsoft, admitían este enfoque de servidor de archivos, y cada computadora personal ejecutaba su propia copia del software DBMS.



Arquitectura cliente / servidor

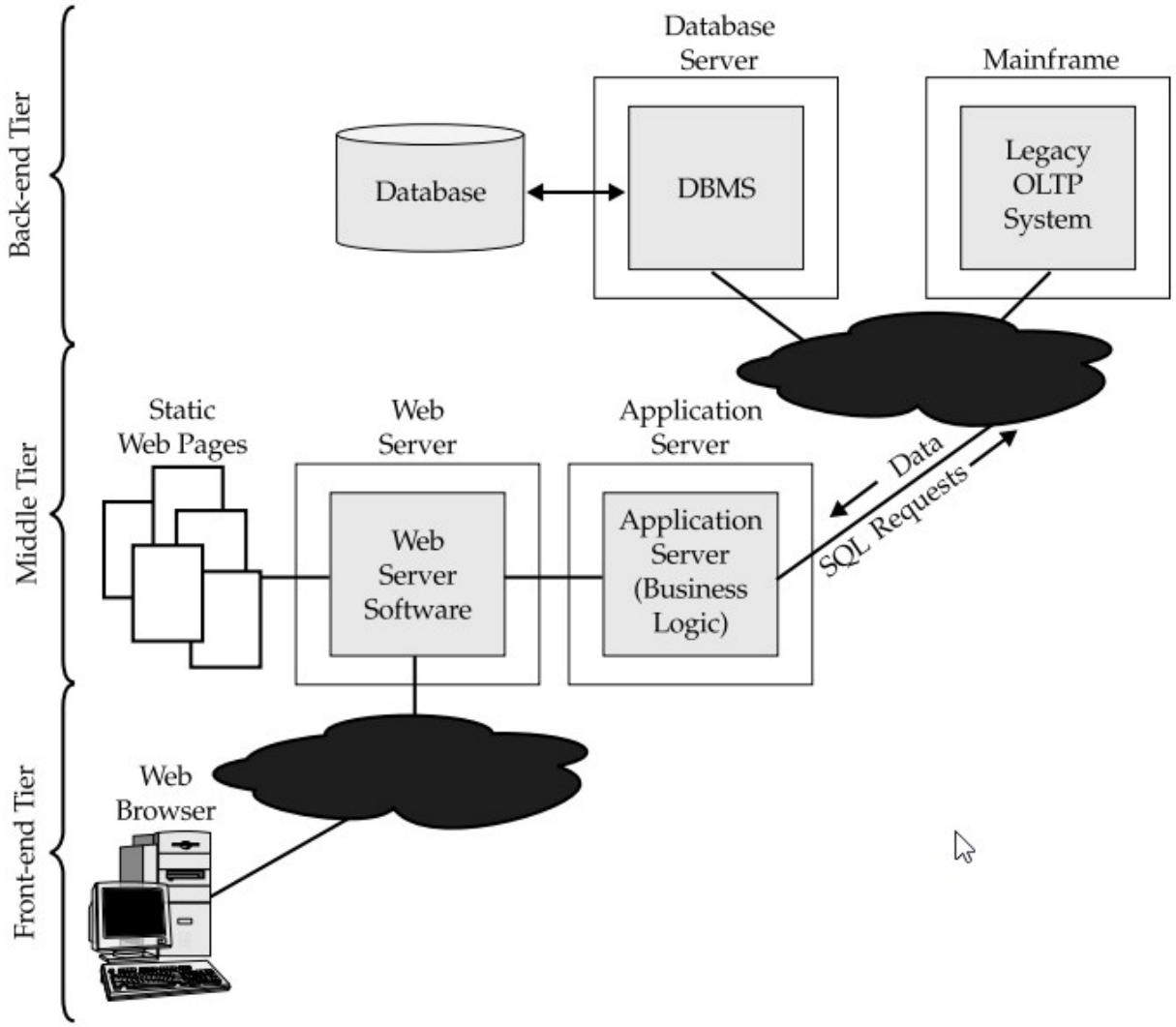
En este esquema, las computadoras personales se combinan en una red de área local con un servidor de base de datos que almacena bases de datos compartidas. Las funciones del DBMS se dividen en dos partes. Las interfaces de base de datos, como las herramientas de consulta interactivas, los redactores de informes y los programas de aplicación, se ejecutan en la computadora personal. El motor de base de datos back-end que almacena y administra los datos se ejecuta en el servidor. A medida que la arquitectura cliente / servidor creció en popularidad durante la década de 1990, SQL se convirtió en el lenguaje de base de datos estándar para la comunicación entre las herramientas de front-end y el motor de back-end en esta arquitectura. La arquitectura cliente / servidor reduce el tráfico de la red y divide la carga de trabajo de la base de datos. Las funciones intensivas para el usuario, como el manejo de entrada y la visualización de datos, se concentran en la PC del usuario.



Arquitectura de varios niveles

Con la aparición de Internet y especialmente de la World Wide Web, Al principio, la Web se utilizó para acceder (examinar) documentos estáticos y evolucionó fuera del mundo de las bases de datos. Pero a medida que se generalizó el uso de navegadores web, no pasó mucho tiempo antes de que las empresas pensaran en usarlos como una forma sencilla de proporcionar acceso a las bases de datos corporativas, Los métodos utilizados para vincular servidores web y sistemas DBMS evolucionaron rápidamente a fines de la

década de 1990 y principios de la de 2000, y han convergido en la arquitectura de red de tres niveles. La interfaz de usuario es un navegador web que se ejecuta en una PC o algún otro dispositivo de cliente ligero, como un teléfono inteligente, en el nivel de front-end. Se comunica con un servidor web en el nivel medio. Todos los productos de servidor de aplicaciones empaquetados proporcionan una API invocable basada en SQL para el acceso a la base de datos. Dado que gran parte del mercado de servidores de aplicaciones ha convergido en torno al estándar Java2 Enterprise Edition (J2EE), Java Database Connectivity (JDBC) se ha convertido en la API estándar líder para el acceso del servidor de aplicaciones a las bases de datos.



La proliferación de SQL

Procesamiento de transacciones y SQL

Las bases de datos SQL y relacionales originalmente tenían muy poco impacto en las aplicaciones de procesamiento de transacciones en línea (OLTP). Con su énfasis en las consultas, las bases de datos relacionales se limitaron al soporte de decisiones y las aplicaciones en línea de bajo volumen, donde su rendimiento más lento no era una desventaja. Para las aplicaciones OLTP, donde cientos de usuarios necesitaban acceso en línea a los datos y tiempos de respuesta de menos de un segundo, el Sistema de Gestión de la Información (IMS) no relacional de IBM reinaba como el DBMS dominante. En 1986, un nuevo proveedor de DBMS, Sybase, introdujo una nueva base de datos basada en SQL especialmente diseñada para aplicaciones OLTP. Sybase DBMS se ejecutó en miniordenadores VAX / VMS y estaciones de trabajo Sun, y se centró en el máximo rendimiento en línea. Oracle Corporation y Relational Technology siguieron pronto

con anuncios de que también ofrecerían versiones OLTP de sus populares sistemas de bases de datos Oracle e Ingres. En el mercado de UNIX, Informix anunció una versión OLTP de su DBMS, llamada Informix-Turbo. En 1988, IBM se subió al tren OLTP relacional con DB2 Versión 2, con pruebas de rendimiento que mostraban que la nueva versión operaba a más de 250 transacciones por segundo en grandes mainframes. IBM afirmó que el rendimiento de DB2 ahora era adecuado para todas las aplicaciones OLTP, excepto las más exigentes, y alentó a los clientes a considerarlo como una alternativa seria a IMS.

A principios de la década de 2000, las bases de datos relacionales basadas en SQL en servidores de bases de datos de alta gama basados en UNIX evolucionaron mucho más allá de la marca de 1000 transacciones por segundo. Los sistemas cliente / servidor que utilizan bases de datos SQL se han convertido en la arquitectura aceptada para implementar aplicaciones OLTP. Desde una posición como "inadecuado para OLTP", SQL ha crecido hasta convertirse en la base estándar de la industria para la creación de aplicaciones OLTP.

Bases de datos SQL y de grupo de trabajo

El espectacular crecimiento de las LAN de PC durante las décadas de 1980 y 1990 creó una nueva oportunidad para la gestión de bases de datos departamentales o de grupos de trabajo. Los sistemas de base de datos originales centrados en este segmento de mercado se ejecutaban en el sistema operativo OS / 2 de IBM. De hecho, SQL Server, ahora una parte clave de la estrategia de Windows de Microsoft, originalmente hizo su debut como un producto de base de datos OS / 2. A mediados de la década de 1990, Novell también hizo un esfuerzo concentrado para hacer de su sistema operativo NetWare una atractiva plataforma de servidor de bases de datos para grupos de trabajo. Desde los primeros días de las LAN de PC, NetWare se había establecido como el sistema operativo de red dominante para servidores de archivos e impresión. Microsoft posicionó con éxito a NT como una plataforma más atractiva para ejecutar aplicaciones de grupos de trabajo (como servidor de aplicaciones) y bases de datos de grupos de trabajo. El propio producto SQL Server de Microsoft se comercializó (y a menudo se incluyó) con NT como una plataforma de base de datos de grupos de trabajo estrechamente integrada.
Hoy en día, SQL está bien establecido como estándar de base de datos para grupos de trabajo. Además de las versiones más recientes de Windows para servidores de Microsoft, Linux se ha convertido en una plataforma muy popular para servidores de grupos de trabajo. Microsoft SQL Server y Oracle comparten la mayor parte del mercado, pero las bases de datos de código abierto como MySQL han surgido como una alternativa muy sólida y rentable. Postgres, otro producto de código abierto desarrollado en la Universidad de California en Berkeley como continuación de Ingres, también ha ganado un número menor pero fiel de seguidores en este segmento.

SQL, almacenamiento de datos e inteligencia empresarial

Durante varios años, el esfuerzo por hacer de SQL una tecnología viable para aplicaciones OLTP desvió el enfoque de las fortalezas originales de las bases de datos relacionales del procesamiento de consultas y la toma de decisiones. Los puntos de referencia de rendimiento y la competencia entre las principales marcas de DBMS se centraron en transacciones simples como agregar un nuevo pedido a la base de datos o determinar el saldo de la cuenta de un cliente. Debido al poder del modelo de base de datos relacional, las bases de datos que las empresas usaban para manejar las operaciones comerciales diarias también podían usarse para analizar la creciente cantidad de datos que se estaban acumulando. Un tema frecuente de conferencias y discursos de ferias comerciales para los gerentes de SI era que los

Los datos (almacenados en bases de datos SQL, por supuesto) deben tratarse como un activo valioso y usarse para ayudar a mejorar la calidad de la toma de decisiones comerciales. Aunque las bases de datos relacionales podrían, en teoría, realizar fácilmente aplicaciones OLTP y de toma de decisiones, existían algunos problemas prácticos muy importantes. Cargas de trabajo OLTP
consistió en muchas transacciones breves de la base de datos, y el tiempo de respuesta para los usuarios fue muy importante. Por el contrario, las consultas de apoyo a la toma de decisiones podrían implicar escaneos secuenciales de tablas de bases de datos grandes para responder preguntas como "¿Cuál es el tamaño promedio de los pedidos por región de ventas?" o "¿Cómo se comparan las tendencias de inventario con la misma época del año anterior?" Estas consultas pueden tardar minutos u horas. Si un analista de negocios intenta ejecutar una de estas consultas durante un momento en que los volúmenes de transacciones comerciales alcanzaron su punto máximo, podría causar una degradación grave en el rendimiento de OLTP. Otro problema fue que los datos para responder preguntas útiles sobre las tendencias comerciales a menudo se distribuían en muchas bases de datos diferentes, por lo general involucrando
diferentes proveedores de DBMS y diferentes plataformas informáticas. El deseo de aprovechar los datos comerciales acumulados y los problemas prácticos de rendimiento que causó en las aplicaciones OLTP llevaron al concepto de almacén de datos, que se muestra en la Figura 3-6. Los datos comerciales se extraen de los sistemas OLTP, se reformatean y validan según sea necesario, y luego se colocan en una base de datos separada que se dedica a consultas de toma de decisiones (el "almacén"). La extracción y transformación de datos se puede programar para el procesamiento por lotes fuera de horario. Idealmente, solo se pueden extraer datos nuevos o modificados, minimizando la cantidad de datos que se procesarán en el ciclo de actualización del almacén mensual, semanal o diario. Con este esquema, las consultas de análisis de negocio que requieren mucho tiempo utilizan el almacén de datos, no la base de datos OLTP, como fuente de datos. Las bases de datos relacionales basadas en SQL fueron una opción clara para el almacén de datos porque
de su procesamiento flexible de consultas. Se formó una serie de nuevas empresas para crear las herramientas de extracción, transformación y consulta de bases de datos que necesita el modelo de almacenamiento de datos. Además, los proveedores de DBMS comenzaron a centrarse en los tipos de consultas de bases de datos que los clientes solían ejecutar en el almacén de datos. Estas consultas tendían a ser grandes y complejas, como analizar decenas o cientos de millones de recibos de caja registradora individuales en busca de patrones de compra de productos. A menudo involucraban datos de series de tiempo, por ejemplo, analizar datos de ventas de productos o participación de mercado a lo largo del tiempo. También tendían a incluir resúmenes estadísticos de datos (ventas totales, volumen promedio de pedidos, crecimiento porcentual, etc.) en lugar de que los propios elementos de datos individuales. Para abordar las necesidades especializadas de las aplicaciones de almacenamiento de datos (a menudo llamadas Procesamiento analítico en línea u OLAP), comenzaron a aparecer bases de datos especializadas. Estas bases de datos se optimizaron para cargas de trabajo OLAP de varias formas diferentes. Su actuación fue ajustada para
acceso a consultas complejas de solo lectura. Admitían funciones estadísticas avanzadas y otras funciones de datos, como el procesamiento integrado de series de tiempo. Apoyaron el cálculo previo de la base de datos
datos estadísticos, por lo que la recuperación de promedios y totales podría ser mucho más rápida. Algunas de estas bases de datos especializadas no usaban SQL, pero muchas sí lo hacían (lo que lleva al término complementario ROLAP, para procesamiento analítico relacional en línea). A medida que el mercado del almacenamiento de datos siguió evolucionando, las herramientas para acceder al almacén surgieron como un

segmento de mercado importante por derecho propio, a menudo denominado inteligencia empresarial. Las líneas entre los proveedores que suministraron las bases de datos del almacén, las herramientas para poblar
ellos, y las herramientas para analizar datos se fueron difuminando gradualmente a medida que crecía el mercado. Tres de los mayores proveedores de inteligencia empresarial se convirtieron en empresas públicas exitosas por derecho propio antes de ser adquiridos por tres de los gigantes de la industria. Business Objects fue adquirido por SAP, el proveedor líder de aplicaciones empresariales. Hyperion fue adquirida por Oracle y Cognos fue adquirida por IBM. Al igual que con tantos segmentos del mercado de TI, las ventajas de SQL como estándar demostraron ser una fuerza poderosa, y los almacenes de datos y las herramientas analíticas basados en SQL están firmemente arraigados.

Aplicaciones de Internet y SQL

A fines de la década de 1990, la World Wide Web y la capacidad de navegación web que habilitó fueron la fuerza impulsora detrás del crecimiento de Internet. Con su enfoque en la entrega de contenido en forma de texto y gráficos, los primeros usos de la Web tuvieron poco que ver con la gestión de datos. Sin embargo, a mediados de la década de 1990, gran parte del contenido entregado desde sitios web corporativos tenía su origen en bases de datos corporativas basadas en SQL. Por ejemplo, en un sitio web comercial para un minorista, las páginas web que contienen información sobre productos disponibles para la venta, sus precios, disponibilidad de productos, promociones especiales y similares se crean normalmente a pedido, basándose en datos recuperados de una base de datos SQL. La gran mayoría de las páginas mostradas por un sitio de subastas en línea o por un sitio de viajes en línea se basan de manera similar en datos recuperado de bases de datos SQL, transformado al formato de página HTML de la Web. En la otra dirección, los datos ingresados por un usuario en los formularios de la página del navegador casi siempre se capturan en bases de datos SQL que forman parte de la arquitectura del sitio web. A principios de la década de 2000, la atención de la industria se había centrado en la siguiente fase de Internet, y
el papel que pueden desempeñar las tecnologías de Internet en la conexión de aplicaciones informáticas entre sí. Estas arquitecturas de aplicaciones distribuidas recibieron una amplia cobertura de la prensa especializada bajo el título de servicios web. En la larga tradición de la industria informática, surgieron campos en competencia que defendían diferentes conjuntos de estándares y lenguajes para implementarlos: un campo dirigido por Microsoft bajo .NET Framework y un campo rival centrado en servidores de aplicaciones basados en Java y J2EE. Ambas arquitecturas adoptan un papel clave para XML, un estándar para el intercambio de datos estructurados como los datos que residen en bases de datos SQL. En respuesta a la atención de la industria en los servicios web, se ha anunciado una serie de productos que vinculan mensajes con formato XML a bases de datos basadas en SQL. Los proveedores de bases de datos de inicio y algunos de los proveedores de bases de datos de objetos anunciaron productos de bases de datos basados en XML, argumentando que proporcionan una combinación nativa ideal para el intercambio de datos en formato XML a través de Internet. Los reproductores de bases de datos relacionales establecidos respondieron con sus propias iniciativas XML, agregando capacidades de entrada / salida XML y luego soporte de tipo de datos XML nativo a sus productos. La integración más estrecha entre XML y SQL sigue siendo un área activa de inversión por parte de todos los principales proveedores de bases de datos en la actualidad. El enfoque de escalabilidad de Internet también está teniendo un gran impacto en los productos de software de bases de datos. Muchos elementos de software de Internet operan a "escala de Internet" mediante un enfoque de escala horizontal, distribuyendo su carga de trabajo entre docenas o cientos de servidores básicos de bajo costo. El motor de búsqueda de Google es uno de los ejemplos más extremos de esta

arquitectura, donde incluso una sola búsqueda puede distribuirse entre docenas de servidores, y el volumen total de búsqueda se distribuye entre decenas de miles de servidores, todos ubicados en la “nube de Internet”. " Existen grandes desafíos para aplicar este enfoque a la gestión de bases de datos, pero proporcionar una gestión de datos "en la nube" es un tema activo de investigación y desarrollo en la comunidad de bases de datos.

Resumen

Este capítulo describió el desarrollo de SQL y su función como lenguaje estándar para la gestión de bases de datos relacionales:

• SQL fue desarrollado originalmente por investigadores de IBM, y el fuerte apoyo de IBM a SQL fue una razón clave para su éxito inicial.

• Existe un estándar SQL ANSI / ISO oficial, que ha crecido enormemente en alcance y complejidad desde su debut en 1986.

• A pesar de la existencia de un estándar, existen muchas pequeñas variaciones entre los dialectos SQL comerciales; no hay dos implementaciones de SQL exactamente iguales.

• SQL se ha convertido en el lenguaje estándar de administración de bases de datos en una amplia gama de sistemas informáticos y áreas de aplicaciones, incluidos mainframes, estaciones de trabajo, computadoras personales, sistemas OLTP, sistemas cliente / servidor, almacenamiento de datos e Internet.