



Compact and High-Speed database for embedded devices

Ubiquitous
DeviceSQL



Overview

Ubiquitous DeviceSQL (hereinafter referred to as "DeviceSQL") appeared in 2002 as a totally new category of data management software specialized for embedded systems and based on stream-based data management technologies. It has been used in cellular phones, onboard equipment, IP set-top boxes, communication equipment and so forth, and the latest version available as of May 2010 was DeviceSQL Release 4.3.

DeviceSQL covers all products, ranging from low-end ones to high-end ones, which cannot be covered by other embedded RDBMSs, and conforms to the Oracle PL/SQL industry-standard programming language. DeviceSQL is capable not only of making inquiries into data using an SQL statement but also of combining an SQL statement with a PL (Procedural Language), such as an IF, FOR or LOOP statement, and thereby performing more complicated processing with a simple command.

DeviceSQL is a next-generation data management framework software (device data management) product specialized for embedded systems that offers the functionality of accessing a database (database functionality) as well as the functionality of combining an SQL statement with procedural language / import function functionality and thereby freely performing data processing (data processing functionality), such as filtering and format conversion, before and after an SQL statement.

► Development environment

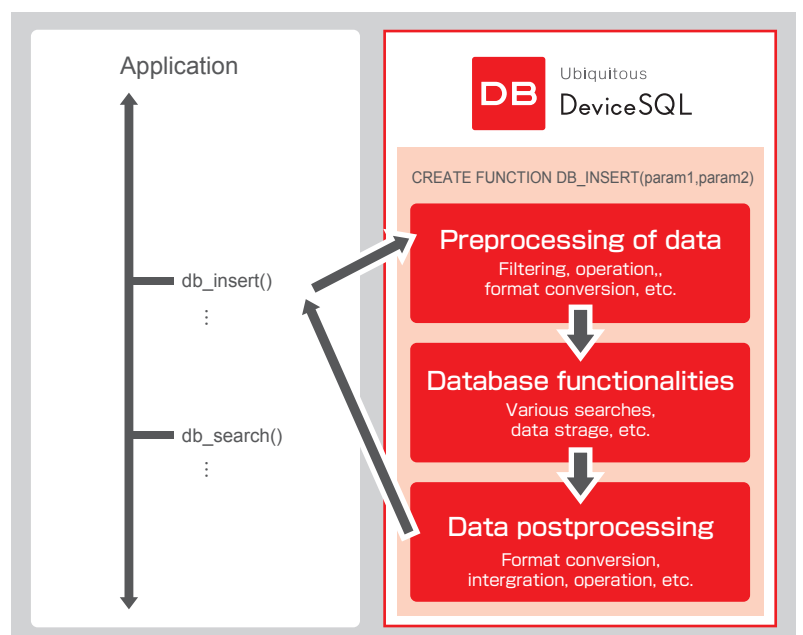
Provided as a framework

DeviceSQL makes it possible to implement the optimum data management system for each embedded system.

Advanced approach from the DeviceSQL Language to the C language

- Enables programming of high productivity through the use of the DeviceSQL high-level language
- Completely separates program control from table handling/data manipulation
- Reduces runtime code size.

DeviceSQL employs the method whereby a programmer describes data logic in the DeviceSQL language and converts that logic into C language code using a compiler, thereby performing during compilation the processing that other embedded RDBMSs generally perform in a runtime environment, such as the interpreting of an SQL statement, syntax checking and error checking. Thanks to this, DeviceSQL achieves a minimal memory footprint and improved performance. In addition, DeviceSQL allows both a function written in C language and a DeviceSQL function to be imported/exported within the DeviceSQL program, which in turn makes detailed data manipulation possible and to completely separate application code and data manipulation code from each other.



【Development environment】

► Platform environment (runtime environment)

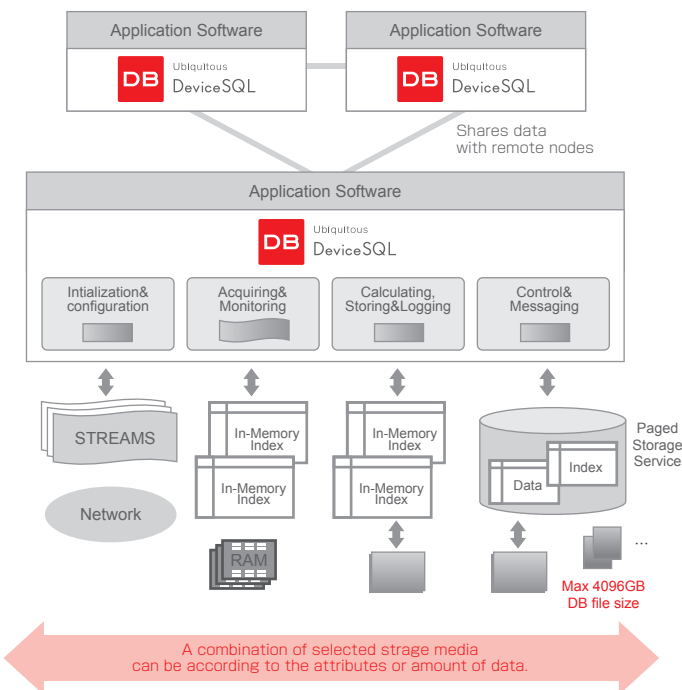
An ultrahigh-speed, ultra compact database

- Is the world's most compact DB engine (a footprint of 24 KB with the minimum configuration and 50 to 80 KB for general use).
- Supports multi-processes and multi-threading.
- Can perform processing at speeds 15 to 50 times higher compared to other ordinary embedded RDBMSs.
- Achieves a stable response time which does not depend on the number of data.
- Supports perfect ACID transactions provided with rollback and crash recovery functionalities.
- Can process various stream data as records.
- Examples of stream data: sensor data, network data, file data
- Supports timeout/interrupt functionalities to be performed in lengthy processing.

Achieving optimal data placement

- Places data at optimal locations by using its in-memory and various permanent storage media (FlashROM, HDD, MMC/SD).
- Contributes to the optimization of performance and reduced power consumption.
- Supports remote data access (share).

Whereas other embedded RDBMSs support either in-memories or permanent storage media, as shown in the following figure DeviceSQL supports four data reference (store) modes: namely, streaming, in-memory, memory-mapped storage and device direct storage (paged storage) modes.



■ Dynamic opening/closing of database files

DeviceSQL allows applications to dynamically open/close a database file at any time. This functionality makes it possible to store message data on a language-by-language basis when an application requires multi-language support and easily switch, for example, from a Japanese message to an English one during the control of the application. In addition, this functionality also makes it possible to temporarily close the database file and free memory resources to make them available to other applications when there are any constraints on memory available in an embedded system and thereby enable the entire system to effectively use that memory.

■ Support for data streams, ideal for real-time processing

DeviceSQL was designed from the beginning of its development with consideration given to making the data management functionalities that are applied by embedded RDBMSs to tables also available to any data source, such as sensor outputs, data sources on a network, or the results of an operation performed by a function. DeviceSQL can perceive input stream data as records, immediately perform a search in SQL, extract the necessary data in real time and transfer it to an application. In addition, DeviceSQL can perform a search for streaming data without change, and therefore can operate with a smaller amount of memory, without needing any redundant working memory.

Interconnection with enterprise-oriented RDBMSs

DeviceSQL DataSync (available as an option) enables data synchronization between enterprise-oriented RDBMSs and DeviceSQL.

DeviceSQL uses new, unprecedented, technologies to provide a framework to improve the performance, operability and intelligence of embedded systems from the aspect of data management as well as to allow embedded systems to create new added value. At the same time, it also makes it possible to achieve higher development productivity and higher quality.

Ubiquitous Corporation

SHINJUKU FIRST WEST 16F, 1-23-7
NISHI-SHINJUKU, SHINJUKU-KU, TOKYO 160-0023, JAPAN
TEL : +81-3-5908-3451 FAX : +81-3-5908-3452
E-MAIL : sales_info@ubiquitous.co.jp
URL : www.ubiquitous.co.jp