SYBASE®

Sybase® Adaptive Server™ Enterprise
Introducing Sybase Adaptive Server Enterprise for UNIX

Adaptive
Server™

Document ID: 34976-01-1150

September 1997

Copyright Information

Copyright © 1989–1997 by Sybase, Inc. All rights reserved.

Sybase, Inc., 6475 Christie Avenue, Emeryville, CA 94608.

Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Sybase, the Sybase logo, APT-FORMS, Certified SYBASE Professional, Data Workbench, First Impression, InfoMaker, PowerBuilder, Powersoft, Replication Server, S-Designor, SQL Advantage, SQL Debug, SQL SMART, SQL Solutions, Transact-SQL, VisualWriter, and VQL are registered trademarks of Sybase, Inc. Adaptable Windowing Environment, Adaptive Component Architecture, Adaptive Server, Adaptive Server Monitor, ADA Workbench, AnswerBase, Application Manager, AppModeler, APT-Build, APT-Edit, APT-Execute, APT-Library, APT-Translator, APT Workbench, Backup Server, BayCam, Bit-Wise, ClearConnect, Client-Library, Client Services, CodeBank, Column Design, Connection Manager, DataArchitect, Database Analyzer, DataExpress, Data Pipeline, DataWindow, DB-Library, dbQ, Developers Workbench, DirectConnect, Distribution Agent, Distribution Director, Dynamo, Embedded SQL, EMS, Enterprise Client / Server, Enterprise Connect, Enterprise Manager, Enterprise SQL Server Manager, Enterprise Work Architecture, Enterprise Work Designer, Enterprise Work Modeler, EWA, Formula One, Gateway Manager, GeoPoint, ImpactNow, InformationConnect, InstaHelp, InternetBuilder, iScript, Jaguar CTS, jConnect for JDBC, KnowledgeBase, Logical Memory Manager, MainframeConnect, Maintenance Express, MAP, MDI Access Server, MDI Database Gateway, media.splash, MetaWorks, MethodSet, Net-Gateway, NetImpact, Net-Library, ObjectConnect, ObjectCycle, OmniConnect, OmniSQL Access Module, OmniSQL Toolkit, Open Client, Open ClientConnect, Open Client / Server, Open Client / Server Interfaces, Open Gateway, Open Server, Open ServerConnect, Open Solutions, Optima++, PB-Gen, PC APT-Execute, PC DB-Net, PC Net Library, Power++, Power AMC, PowerBuilt, PowerBuilt with PowerBuilder, PowerDesigner, Power J, PowerScript, PowerSite, PowerSocket, Powersoft Portfolio, Power Through Knowledge, PowerWare Desktop, PowerWare Enterprise, ProcessAnalyst, Quickstart Datamart, Replication Agent, Replication Driver, Replication Server Manager, Report-Execute, Report Workbench, Resource Manager, RW-DisplayLib, RW-Library, SAFE, SDF, Secure SQL Server, Secure SQL Toolset, Security Guardian, SKILS, smart.partners, smart.parts, smart.script, SQL Anywhere, SQL Central, SQL Code Checker, SQL Edit, SQL Edit / TPU, SQL Modeler, SQL Remote, SQL Server, SQL Server / CFT, SQL Server / DBM, SQL Server Manager, SQL Server SNMP SubAgent, SQL Station, SQL Toolset, Sybase Client / Server Interfaces, Sybase Development Framework, Sybase Gateways, Sybase IQ, Sybase MPP, Sybase SQL Desktop, Sybase SQL Lifecycle, Sybase SQL Workgroup, Sybase Synergy Program, Sybase Virtual Server Architecture, Sybase User Workbench, SybaseWare, SyBooks, System 10, System 11, the System XI logo, SystemTools, Tabular Data Stream, The Architecture for Change, The Enterprise Client / Server Company, The Model for Client / Server Solutions, The Online Information Center, Translation Toolkit, Turning Imagination Into Reality, Unibom, Unilib, Uninull, Unisep, Unistring, Viewer, Visual Components, VisualSpeller, WarehouseArchitect, WarehouseNow, Warehouse WORKS, Watcom, Watcom SQL, Watcom SQL Server, Web.SQL, WebSights, WebViewer, WorkGroup SQL Server, XA-Library, and XA-Server are trademarks of Sybase, Inc. 6 / 97

All other company and product names used herein may be trademarks or registered trademarks of their respective companies.

Table of Contents

List of Figures

About This Book

This book, *Introducing Sybase Adaptive Server Enterprise for UNIX*, provides an overview of Adaptive Server™ Enterprise for UNIX, a set of Sybase® products for developing and deploying relational database applications on desktop platforms.

Audience

Read this guide before you install and begin using Adaptive Server to establish a foundation for using the product. This guide is for System Administrators, managers, or anyone who will be involved in setting up or using Adaptive Server or who wants to understand how the product set fits together. This guide assumes no technical knowledge of the Sybase products.

How to Use This Guide

This guide includes the following chapters:

- Chapter 1, "Overview of Adaptive Server Enterprise," introduces relational database management systems and client/server architecture. It also describes each of the products in the Adaptive Server set of products.
- Chapter 2, "Terms and Concepts," defines some of the basic terminology necessary for understanding discussions about using Adaptive Server.
- Chapter 3, "Using Adaptive Server Enterprise," provides a brief introduction to some of the most common tasks a user performs with Adaptive Server.

Related Documents

The *Read Me First* card in your product package lists the products that are included with Adaptive Server Enterprise for UNIX. It also provides a road map for locating the related Sybase documentation. *Installing Adaptive Server and OmniConnect on UNIX Platforms* lists the version numbers of the included products.

Conventions

This manual uses the following style conventions:

- Within text, the names of files, directories, and database objects appear in italics:

  /data/master.dat

- Examples showing the use of Transact-SQL commands are printed like this:

  ```
  select titles, pub_id
      from titles
  ```

- The names of utilities, procedures, commands, and scripts appear in the following font:

  sp_revokelogin

- Terms that you can find in the glossary appear in the following font:

  global variable

Adaptive Server Enterprise Documents

The following documents comprise the Sybase Adaptive Server Enterprise documentation:

- The *Release Bulletin* for your platform – contains last-minute information that was too late to be included in the books.

  A more recent version of the *Release Bulletin* may be available on the World Wide Web. To check for critical product or document information that was added after the release of the product CD, use SyBooks™-on-the-Web.

- The Adaptive Server installation documentation for your platform – describes installation and upgrade procedures for all Adaptive Server and related Sybase products.

- The Adaptive Server configuration documentation for your platform – describes configuring a server, creating network connections, configuring for optional functionality, such as auditing, installing most optional system databases, and performing operating system administration tasks.

- *What's New in Adaptive Server Enterprise?* – describes the new features in Adaptive Server release 11.5, the system changes added to support those features, and the changes that may affect your existing applications.

- *Navigating the Documentation for Adaptive Server* – an electronic interface for using Adaptive Server. This online document provides links to the concepts and syntax in the documentation that are relevant to each task.
- *Transact-SQL User's Guide* – documents Transact-SQL, Sybase's enhanced version of the relational database language. This manual serves as a textbook for beginning users of the database management system. This manual also contains descriptions of the *pubs2* and *pubs3* sample databases.
- *System Administration Guide* – provides in-depth information about administering servers and databases. This manual includes instructions and guidelines for managing physical resources and user and system databases, and specifying character conversion, international language, and sort order settings.
- *Adaptive Server Reference Manual* – contains detailed information about all Transact-SQL commands, functions, procedures, and datatypes. This manual also contains a list of the Transact-SQL reserved words and definitions of system tables.
- *Performance and Tuning Guide* – explains how to tune Adaptive Server for maximum performance. This manual includes information about database design issues that affect performance, query optimization, how to tune Adaptive Server for very large databases, disk and cache issues, and the effects of locking and cursors on performance.
- The *Utility Programs* manual for your platform – documents the Adaptive Server utility programs, such as **isql** and **bcp**, which are executed at the operating system level.
- *Security Administration Guide* – explains how to use the security features provided by Adaptive Server to control user access to data. This manual includes information about how to add users to Adaptive Server, administer both system and user-defined roles, grant database access to users, and manage remote Adaptive Servers.
- *Security Features User's Guide* – provides instructions and guidelines for using the security options provided in Adaptive Server from the perspective of the non-administrative user.

- *Error Messages* and *Troubleshooting Guide* – explains how to resolve frequently occurring error messages and describes solutions to system problems frequently encountered by users.
- *Component Integration Services User's Guide for Adaptive Server Enterprise and OmniConnect* – explains how to use the Adaptive Server Component Integration Services feature to connect remote Sybase and non-Sybase databases.
- *Adaptive Server Glossary* – defines technical terms used in the Adaptive Server documentation.
- *Master Index for Adaptive Server Publications* – combines the indexes of the *Adaptive Server Reference Manual*, *Component Integration Services User's Guide*, *Performance and Tuning Guide*, *Security Administration Guide*, *Security Features User's Guide*, *System Administration Guide*, and *Transact-SQL User's Guide*.

Other Sources of Information

Use the SyBooks™ and SyBooks-on-the-Web online resources to learn more about your product:

- SyBooks documentation is on the CD that comes with your software. The DynaText browser, also included on the CD, allows you to access technical information about your product in an easy-to-use format.

  Refer to *Installing SyBooks* in your documentation package for instructions on installing and starting SyBooks.
- SyBooks-on-the-Web is an HTML version of SyBooks that you can access using a standard Web browser.

  To use SyBooks-on-the-Web, go to http://www.sybase.com, and choose Documentation.

If You Need Help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

1 Overview of Adaptive Server Enterprise

Adaptive Server Enterprise is an integrated set of software products for designing, developing and deploying relational database applications. It consists of a high-performance relational database management system (RDBMS), which runs database servers, and a collection of applications and libraries, which run on database clients. This arrangement, consisting of servers that are accessed by multiple clients over a network, forms the basis for Sybase's **client/server architecture**.

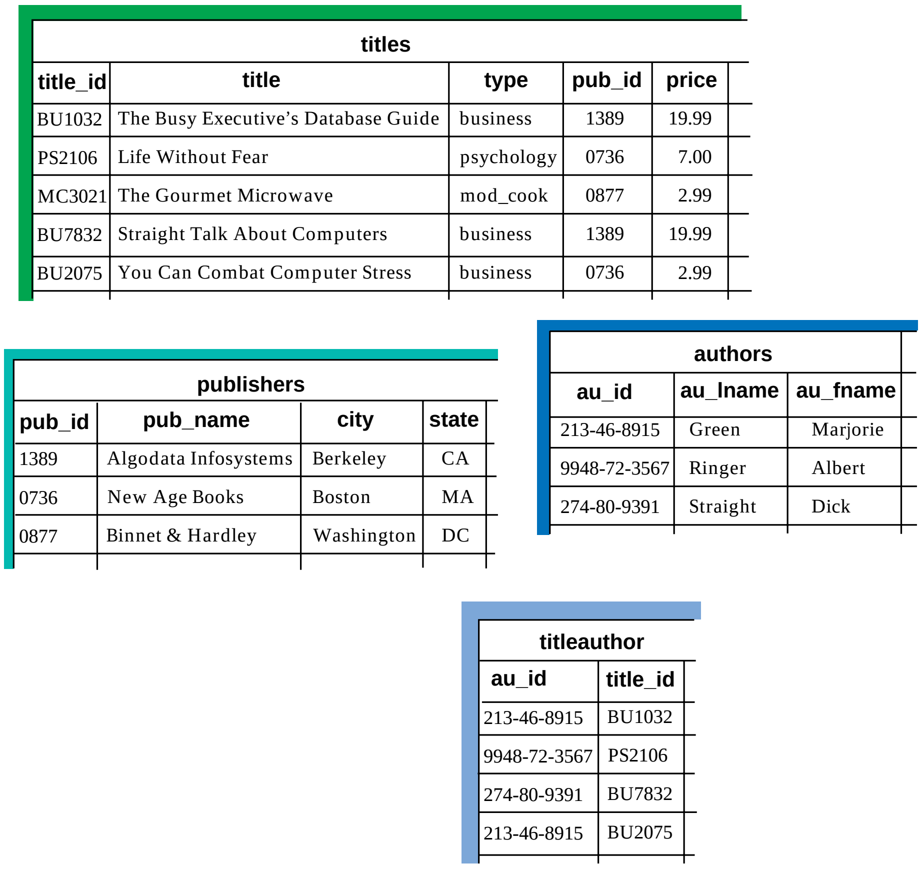
This chapter provides an overview of the following topics:

Relational Database Management Systems

A **relational database management system** (**RDBMS**) is a system for storing and retrieving data in which the data is represented in two-dimensional **tables**. In early relational systems, tables were called relations. A relational database consists of a collection of tables that store interrelated data.

Tables

Figure 1-1 shows portions of tables that you might create in a relational database that stores related data about books: The tables are called *titles, authors, publishers,* and *titleauthor.*



titles

title_id	title	type	pub_id	price
BU1032	The Busy Executive's Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

publishers

pub_id	pub_name	city	state
1389	Algodata Infosystems	Berkeley	CA
0736	New Age Books	Boston	MA
0877	Binnet & Hardley	Washington	DC

authors

au_id	au_lname	au_fname
213-46-8915	Green	Marjorie
9948-72-3567	Ringer	Albert
274-80-9391	Straight	Dick

titleauthor

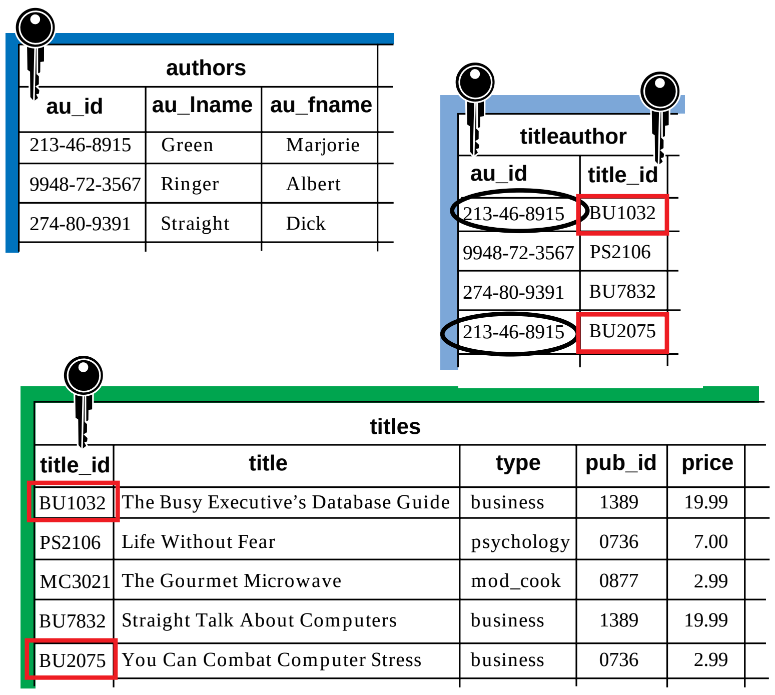
au_id	title_id
213-46-8915	BU1032
9948-72-3567	PS2106
274-80-9391	BU7832
213-46-8915	BU2075

Figure 1-1: Tables in a database

Each table in the database holds information about different things—books, publishers, and authors. But some information in each table overlaps with information in another table—by design. Columns that appear in more than one table and that link the tables together are called **keys**. Keys, which are discussed in more detail in "Key" on page 2-2, allow you to retrieve information that is distributed over multiple tables.

For example, you might want information about books published by Marjorie Greene, which spans the *authors* and the *titles* tables. Figure

1-2 shows the key columns for the sample databases and how they allow you to retrieve the correct data from the tables.



authors

au_id	au_lname	au_fname
213-46-8915	Green	Marjorie
9948-72-3567	Ringer	Albert
274-80-9391	Straight	Dick

titleauthor

au_id	title_id
213-46-8915	BU1032
9948-72-3567	PS2106
274-80-9391	BU7832
213-46-8915	BU2075

titles

title_id	title	type	pub_id	price
BU1032	The Busy Executive’s Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

Figure 1-2: Retrieving data across tables using key columns

One advantage of relational databases is that they allow you to set up multiple tables, a structure that eliminates redundancy and possible inconsistencies caused by redundancy. For example, both the sales and accounts payable departments might enter and look up information about publishers. In a relational database, the information about publishers is entered only once, in a table that both departments can access.

Defining how tables represent a body of information, determining how the data should be distributed among the tables, and deciding what keys are needed to define the relationships are the subjects of **logical design**. Sound database design is the key factor in realizing Adaptive Server’s performance potential and makes a database easy to maintain and update. Many excellent books are available on this subject. *The Practical SQL Handbook* (Bowman, Emerson, and Damovsky, Addison-Wesley, 1993) offers a clear introduction to logical database design, and it specifically relates the issues to an Adaptive Server context. (See “Designing Databases” on page 3-2.)

SQL

SQL (Structured Query Language) is the database language designed for the RDBMS model. It is used to set up, use, and maintain a relational database. This is in contrast to the database systems in previous computing models, which required large Information Services departments to be involved in data requests. An excellent reference for learning SQL is the *LAN Times Guide to SQL* (Groff and Weinberg, Osborne McGraw-Hill, 1994).

End users need not learn SQL in order to access data—in most systems, purchased or developed GUI client software translates user requests into SQL. Application programming interfaces (APIs) or libraries installed on the clients allow applications to send SQL commands to and retrieve results from the RDBMS.

Sybase’s enhanced version of SQL is called **Transact-SQL** (see “Transact-SQL” on page 2-6).

Auxiliary Facilities of Adaptive Server’s RDBMS

In addition to storing and retrieving data, Adaptive Server provides a number of auxiliary facilities for managing the relational databases. These include programs for starting and stopping the server, for backing up and restoring data, for copying data between a database and a file system, for authenticating users of the server and the databases, for managing the physical devices on which the data is stored, for displaying system messages in different languages, for storing and preprocessing a reusable procedure or set of procedures, and so forth. Together with the basic storage and retrieval capabilities, these facilities make up Adaptive Server’s relational database management system.

Client/Server Architecture

In a Sybase Adaptive Server environment, the databases and the RDBMS software reside on one or more servers, where multiple clients can access them concurrently over a network. Applications that access and manipulate the shared data, as well as the libraries that allow the applications to communicate with the RDBMS, reside on the clients. Figure 1-3 illustrates the Sybase client / server configuration.

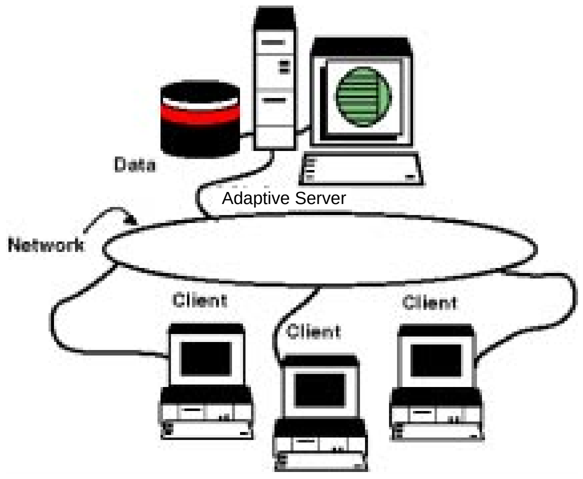


Figure 1-3: Sybase client/server environment

The basic client / server relationship is one in which client applications request services from servers, and servers return results to client applications. A typical request for service in a client / server RDBMS is for the client to request that the server retrieve or modify the data that it is storing. A typical result is confirmation by the server that the request was successfully or unsuccessfully carried out, followed by the requested data. Figure 1-4 illustrates a basic client / server interaction.

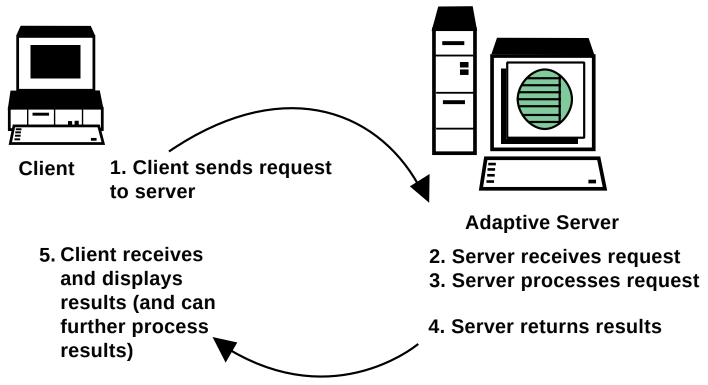

Client
1. Client sends request to server
Adaptive Server
2. Server receives request
3. Server processes request
4. Server returns results
5. Client receives and displays results (and can further process results)

Figure 1-4: Basic client/server interaction

Division of Tasks

A client / server model provides the best performance for data sharing among a large number of PCs and workstations because clients and servers share the work—each performs the work it can do most efficiently. The server handles activities related directly to the maintenance and retrieval of shared data, while other tasks, such as displaying returned data or interpreting requests from users, are off-loaded to the clients. Figure 1-5 represents how the tasks are divided.

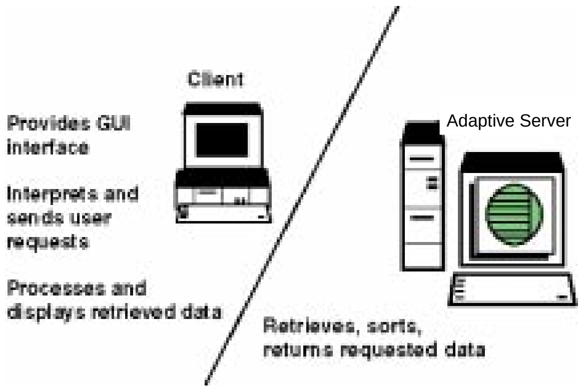

Client
Provides GUI interface
Interprets and sends user requests
Processes and displays retrieved data
Adaptive Server
Retrieves, sorts, returns requested data

Figure 1-5: Division of tasks

Distributed Computing

The ability to distribute data among multiple servers provides a structure for distributed computing environments in which control is not centralized at a single data source or geographic location. Clients can access the data they need from any server.

Figure 1-6 shows a noncentralized, distributed computing configuration in which data is distributed over several sites; each site houses a different subset of the data.

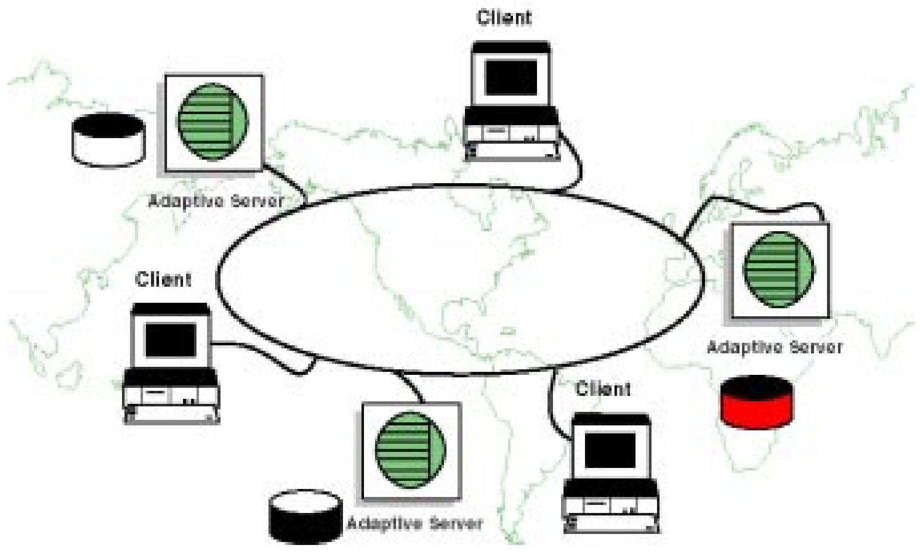


Figure 1-6: Distributed client/server environment

Some organizations want each site to have a complete copy of the entire data set to improve the time it takes to access the data and to eliminate downtime in the case of one server's failure. Data **replication** offers a solution to these organizations.

Replication Server®, a Sybase product that you can purchase separately, can be integrated with the Adaptive Server Enterprise setup. Replication Server replicates peer-to-peer data replication across a distributed environment, to and from heterogeneous hardware and data sources. All clients have equally fast and reliable access to all data.

Components of Adaptive Server Enterprise

As described in "Client / Server Architecture" on page 1-4 , the basic client / server RDBMS model consists of the database server and third-party or developed client software that communicates with the database server over a network. Adaptive Server Enterprise also includes software for backing up, monitoring, and administering Adaptive Server; client libraries that enable applications to communicate with the server; and developed client applications that can meet many of your computing needs.

You install some Adaptive Server Enterprise software on the server and some on the client. It is also convenient to run some of the client administrative tools and utilities from the server desktop.

On the server, you create files for use as **database devices**, files dedicated to storing data. Use the Transact-SQL **disk init** command to initialize the devices.

On the client, you can install (in addition to the client components of Adaptive Server) **development tools** such as PowerBuilder®, that help you build applications that access Adaptive Server, the applications themselves, and / or third-party software.

Figure 1-7 shows an overview of the client and server components of an Adaptive Server system.

The sections "Server Components" on page 1-8 and "Client Components" on page 1-9 describe each component in more detail:

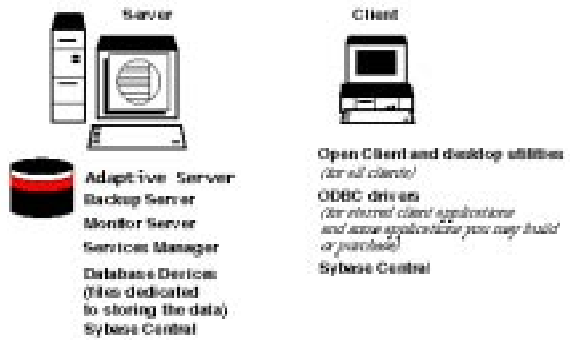


Figure 1-7: Overview of Adaptive Server Enterprise components

Server Components

The following Adaptive Server Enterprise components reside on the server:

- **Adaptive Server**

  Adaptive Server is Sybase's high-performance RDBMS. See *Understanding Sybase SQL Server 11* for a technical overview of Adaptive Server release 11.5.x, and the white papers accessible from the Sybase home page on the World Wide Web at the following address:

http://www.sybase.com

Adaptive Server also includes several utilities, which are described in *Utility Programs for UNIX*.

- **Backup Server**™

  Backup Server, is a server application that runs concurrently with Adaptive Server to perform high-speed on-line database **dumps** and **loads**. Backup Server is automatically installed when you install Adaptive Server.

- **Server Components of Adaptive Server Monitor™**

  Monitor Server and **Monitor Historical Server** are the server components of a client/server application called Adaptive Server Monitor, which allows you to capture, display, and evaluate Adaptive Server performance data and to tune Adaptive Server performance. Monitor Server captures performance data from Adaptive Server's shared memory; Monitor Historical Server, writes the data to files for offline analysis. You must install Monitor Server on the same machine as the Adaptive Server that you want to monitor. Monitor Historical Server performs best when installed on a different machine from the Adaptive Server being monitored.

- **Component Integration Services**

  Component Integration Services extends Adaptive Server and provides enhanced interoperability. It allows Adaptive Server to present a uniform view of enterprise data to client applications and provides location transparency to enterprise-wide data sources.

Client Components

When you install client products on a PC, the installation procedure creates a Sybase program group. This program group contains entries for many of the client components.

The following Adaptive Server Enterprise components are client components:

- **Sybase Central**™

  Sybase Central is a Windows application for managing Sybase databases. It helps you manage database objects and perform common administrative tasks. It makes complex tasks (such as backing up a database) easy.

Sybase Central helps with the following typical system or database administrator’s tasks:

- Managing Adaptive Servers, including connecting to, disconnecting from and stopping servers, and troubleshooting Adaptive Server problems
- Managing **data caches**
- Managing Adaptive Server physical resources
- Managing databases, including creating, deleting, backing up, and restoring databases
- Managing access, including creating and deleting Adaptive Server **logins**, creating and deleting database users and user **groups**, administering Sybase **roles**, and managing object and command **permissions**
- Monitoring Adaptive Server performance data and tuning Adaptive Server performance parameters

• **Open Client**

Open Client contains APIs that enable client applications to interface with Adaptive Server. These APIs are Client-Library™, DB-Library™, and CS-Library. Client applications that you purchase or develop, as well as application development tools such as PowerBuilder, require that these Open Client libraries be installed so that they can communicate with Adaptive Server.

CS-Library contains a collection of utility routines used by all client applications. Client-Library and DB-Library contain a collection of routines that are specific to the programming language being used in an application. Client-Library is an extension of the older DB-Library that accommodates new features supported by Sybase SQL Server releases 10.0 and later; DB-Library supports pre-release 10.0 features. For more information, see the documentation in the Open Client / Server™ collection of SyBooks™ (see “Sybase Documentation” on page 1-11 for more information).

Open Client includes Net-Library™, a library containing network protocol services that support connections between client applications and Adaptive Server.

Open Client also includes the following utilities:

- isql – an interactive query processor that lets you send commands to the RDBMS from the command line.

- **bcp** – a program that copies data from a database to an **operating system file**, and vice versa.
- **defncopy** – a program that copies definitions of **database objects** that you create from a database to an operating system file and vice-versa.

You install Open Client on all machines that will run client software that accesses Adaptive Server.

- **Desktop Utilities**

  Icons for frequently used client utilities appear on the desktop. These include:

  - **dsedit** – an editor for creating and modifying the network configuration files. You can also call **dsedit** from Sybase Central. **dsedit** includes a component called **sybping** that tests network connections between client PCs and a server. **wdllvers** (Windows only) – a utility for examining Sybase and Windows Dynamic Link Libraries (**DLLs**) loaded into memory.
  - **SQL Advantage**® – a utility for executing Transact-SQL® commands and system procedures using a GUI interface.

- **Open Database Connectivity (ODBC) Drivers**

  ODBC is an API developed by Microsoft to allow clients to connect to heterogeneous RDBMSs.

  You should also install the drivers on clients where you will be running any third-party products that require the ODBC interface. You need to install Open Client software in addition to the ODBC drivers in order to connect properly.

  After you install the ODBC drivers, you create an **ODBC data source** for Adaptive Server in the ODBC Administrator. The ODBC Administrator is a utility that appears on the desktop after you install the drivers. Then you connect through the ODBC interface from the client application.

Sybase Documentation

Except for the books and online help files included in the Adaptive Server Enterprise package, all the product documentation is contained in an online library called SyBooks, which is included on the product media. You can install SyBooks on a single server, where users can access it over the network, or you can install it on individual clients, if they have enough disk space.

If you have Internet access and a Web browser, you can also access SyBooks from the World Wide Web. See "Other Sources of Information" on page xii for more information about SyBooks documentation.

You can also order the documentation in the SyBooks collection in printed form. Refer to the product ID numbers listed on the SyBooks content list included in the product package.

2 Terms and Concepts

Before you delve into Adaptive Server and Adaptive Server documentation, familiarize yourself with some of the terminology that you will frequently encounter as you explore the Sybase family of relational database products.

Chapter 1 defined some terms, including **client/server architecture**, **relational database**, **relational database management system** (RDBMS), and **table**.

This chapter describes some additional terms and concepts that relate to Adaptive Server, including:

Database Object

The term **database object** refers to a component of a relational database. The primary database object is the **table**. This section describes other database objects: **rows** and **columns**, **defaults**, **views**, **indexes**, **rules**, **triggers**, and **stored procedures**.

Row and Column

A table contains information about an entity. For example, a table might contain information about authors. A **row**, or record, describes one instance of that entity—a person, a company, a sale, or some other entity. The following example describes a particular author. A **column**, or field, describes an attribute of that entity, such as name, size, color, price, and so on. The example in Figure 2-1 shows columns that represent the author’s first and last names. In early relational systems, columns were called attributes.

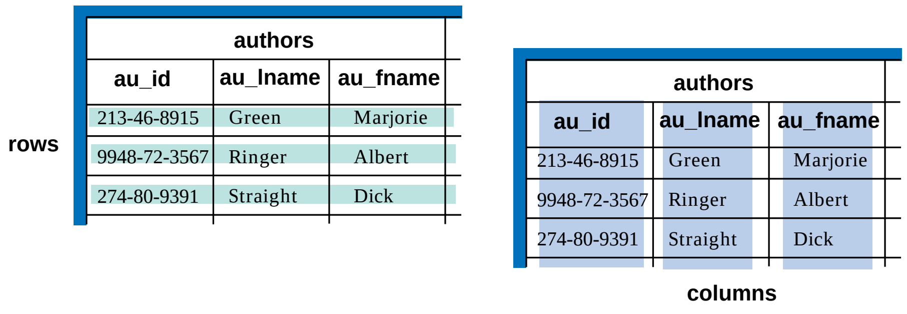
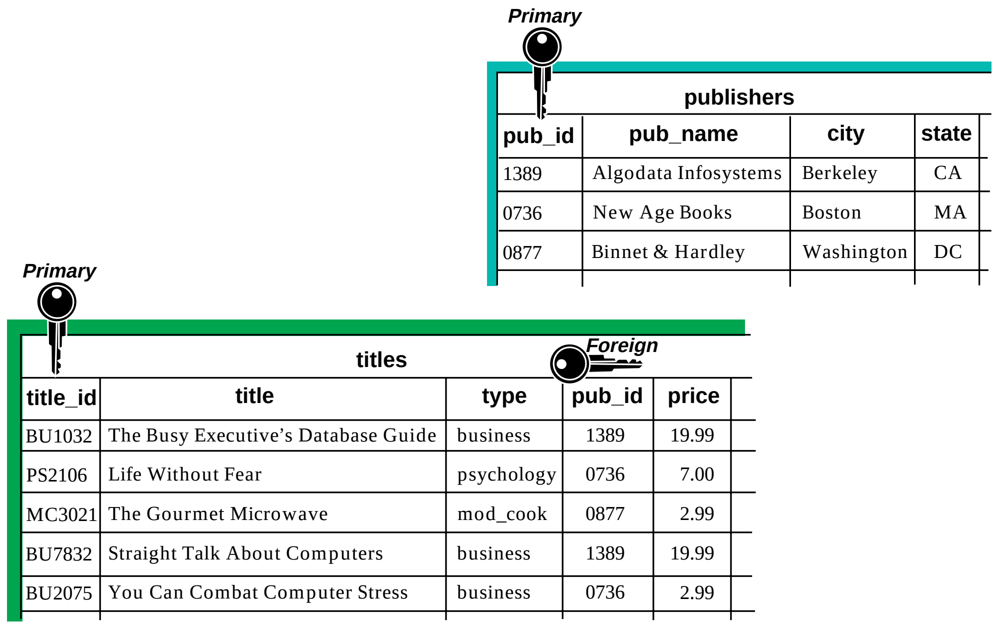


Figure 2-1: Rows and columns in a table

Key

In some cases, a column does not describe a real-world attribute but, instead, is a **key** column that serves a logical function, (as introduced in “Tables” on page 1-1). A **primary key** uniquely identifies a row in a table. A **foreign key** relates a row in one table to a row in another table by matching its value to the value of the other table’s primary key.

The example shown in Chapter 1 shows three key columns. The *pub_id* column in the *publishers* table is a primary key that provides a unique identifier for every row in the *publishers* table—there are no duplicated *pub_id* values. In the *titles* table, however, the *pub_id* column is a foreign key that relates the information about a book in the *titles* table to information about that book’s publisher in the *publishers* table. In the *titles* table, there may be *pub_id*s that appear more than once, because a publisher probably publishes more than one book.



publishers

pub_id	pub_name	city	state
1389	Algodata Infosystems	Berkeley	CA
0736	New Age Books	Boston	MA
0877	Binnet & Hardley	Washington	DC

titles

title_id	title	type	pub_id	price
BU1032	The Busy Executive’s Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

Figure 2-2: Primary and foreign key columns

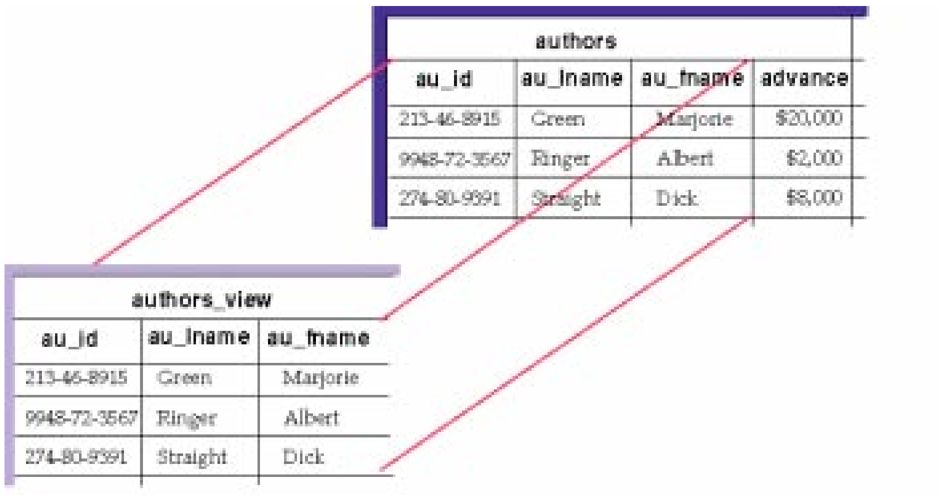
You must enforce **referential integrity** for foreign keys. In other words, for every foreign key that does not have a NULL value, there must be a corresponding primary key with the same value. In the example in Figure 2-2, this means that if a *pub_id* in a row in the *titles* table (the “referencing” table) has a value of 1389, there must a row with a *pub_id* primary key value of 1389 in the *publishers* table (the referenced table). You can enforce referential integrity by specifying **constraints** when you create the table or you can enforce it with **triggers** (see “Trigger” on page 2-6).

A **default** is the value that Adaptive Server assigns to a column when a row is added to a table specifying no particular value for that column. You can specify a default for a column when you create a table. For example, for a column that identifies the price of a published book, you can define “0.00” as the default value until a price is assigned to the book.

View

A **view** is a virtual table, composed of specific columns and / or rows from one or more tables or other views. A view is virtual because it has no physical existence in the database.

You can use a view to simplify and customize a user's perception of a database. For example, sales clerks might need to access some data about authors, but they should not have access to confidential information about an author's royalty advances. You can create a view that customizes the way sales clerks can access the database. Figure gives an idea of how a view can select information from a table or tables:



authors

au_id	au_lname	au_fname	advance
213-46-8915	Green	Marjorie	$20,000
9948-72-3567	Ringer	Albert	$2,000
274-80-9391	Straight	Dick	$8,000

authors_view

au_id	au_lname	au_fname
213-46-8915	Green	Marjorie
9948-72-3567	Ringer	Albert
274-80-9391	Straight	Dick

Figure 2-3: A view

Different views can present combinations of data from various tables in different ways, independent of the table structure in which the data is stored. In some cases, data can be directly updated through a view; in other cases, a view is read-only.

Index

An **index** is a database object that helps Adaptive Server locate data. Indexes speed up data retrieval by pointing Adaptive Server to the location of a table column's data on disk.

Conceptually, an index in a database is like an index in a book. In a book, the index relates each indexed term to the page or pages on which that word appears. In a database, the index relates each indexed column value to the **page**, or physical location, at which the row of data containing the indexed value is stored. See Figure 2-4.

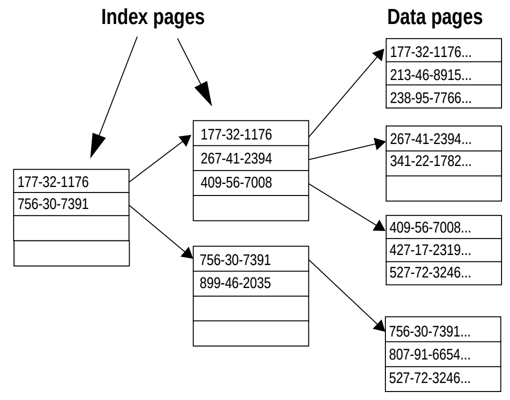


Figure 2-4: A simplified index schematic

Indexes are an important physical design element for high performance. The *Performance and Tuning Guide* explains in detail how indexes work and how to create indexes that improve performance.

Rule

A **rule** defines and enforces which data can be entered for a particular table column or user-defined datatype. For example, you can define a rule that requires that all the values in the *title_id* column in the *titles* table begin with two uppercase letters followed by four digits. Or you can define a rule that requires all values of the *money* datatype to be less than $1,000,000.

Stored Procedure

A **stored procedure** is a set of one or more commands stored in a database. A stored procedure is partially processed before it is stored, so it executes faster than it would if you executed its constituent commands individually.

A stored procedure is invoked by its name. The caller can pass **parameters** to and receive results from stored procedure. You can create and name your own stored procedures to execute specific database queries and to perform other database tasks. For example, you might create a stored procedure that returns the names of all authors whose books have sold more than the number of books that you specify as a parameter at the time you call the procedure.

You can also use a set of stored procedures supplied by Sybase to help you accomplish numerous administrative tasks. These Sybase-supplied stored procedures are called **system procedures** and are automatically installed when you install Adaptive Server. The system procedures are described in the *Adaptive Server Reference Manual*.

An **extended stored procedure** (**ESP**) is a stored procedure that executes procedural language code instead of database commands. You invoke the extended stored procedure in the same way as a regular stored procedure. A Sybase-supplied extended stored procedure is called a system extended stored procedure or a system ESP. The system ESPs are described in the *Adaptive Server Reference Manual*.

Trigger

A **trigger** is a special stored procedure that automatically fires whenever a user updates, deletes, or inserts data, depending on how you define the trigger. You associate a trigger with a table or columns within a table.

For example, some organizations keep duplicate data to improve retrieval speed for **decision support** queries. They can define triggers to duplicate data automatically after an update, deletion, and insertion. Managers can query duplicated decision support data without slowing down the data entry system. Another common use of a trigger is to implement some aspects of **referential integrity**.

Transact-SQL

Transact-SQL is Sybase's enhanced version of SQL. It is the language that you, or the client application you are running, uses to communicate requests to the RDBMS. Some of the enhancements that Transact-SQL offers, such as **stored procedures**, **control-of-flow language**, and **error handling** allow Adaptive Server to be a truly programmable database server.

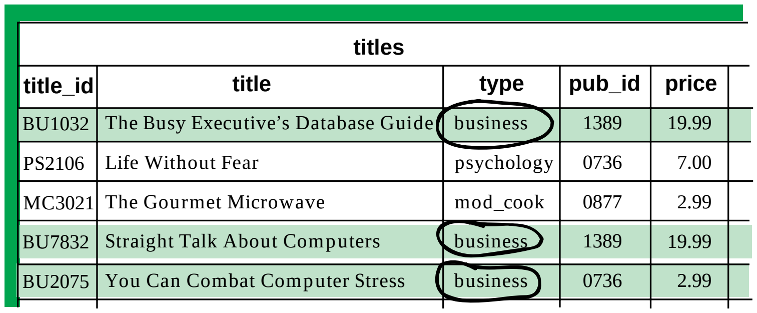
Queries

The “Q” in “SQL” stands for **query**. You query or retrieve data from a database with Transact-SQL’s **select** command. The basic query operations in a relational system are **selection**, **projection**, and **join**. The **select** command implements all of them.

The following sections present some simplified scenarios of retrieving data from a database using the **select** command and show the actual SQL statements you use to accomplish the queries. The purpose of these statements is to familiarize you with what Transact-SQL looks like, not to teach you SQL. For detailed instructions in how to use Transact-SQL, see the *Transact-SQL User’s Guide*.

Selection

A **selection** (also called restriction) is a subset of the rows in a table, based on some conditions. Figure 2-5 shows an example of selecting the rows for books of the type “business” from the *titles* table.



titles				
title_id	title	type	pub_id	price
BU1032	The Busy Executive’s Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

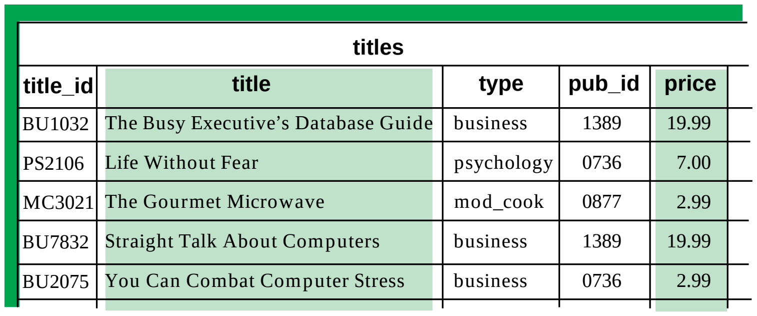
Figure 2-5: Selection

The Transact-SQL **select** command that retrieves the highlighted rows is:

```
select *
    from titles
    where type like “business”
```

Projection

A **projection** is a subset of the columns in a table. Figure 2-6 shows a sample projection in which you want to look only at the title and price of all books in the *titles* table.



titles

title_id	title	type	pub_id	price
BU1032	The Busy Executive’s Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

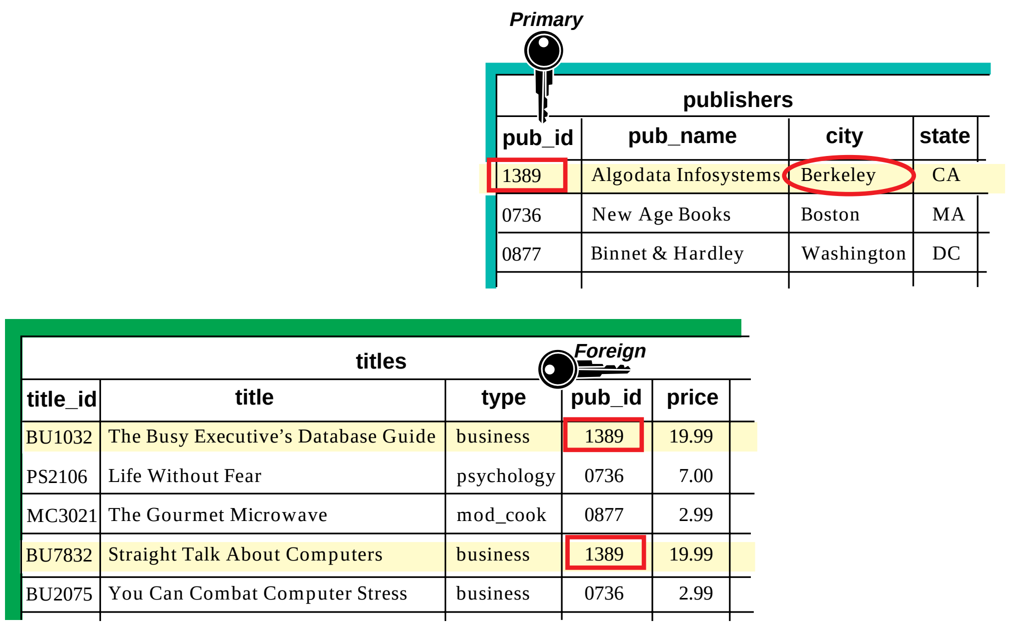
Figure 2-6: Projection

The Transact-SQL **select** command that retrieves the highlighted columns is:

```
select titles, pub_id
    from titles
```

Join

A **join** links the rows in two or more tables by comparing the values in key columns and concatenating rows that have matching values. For example, you might want to select data from the relevant tables showing information about Berkeley publishers and the types and titles of books they published. Figure 2-7 shows a join.



publishers

pub_id	pub_name	city	state
1389	Algodata Infosystems	Berkeley	CA
0736	New Age Books	Boston	MA
0877	Binnet & Hardley	Washington	DC

titles

title_id	title	type	pub_id	price
BU1032	The Busy Executive’s Database Guide	business	1389	19.99
PS2106	Life Without Fear	psychology	0736	7.00
MC3021	The Gourmet Microwave	mod_cook	0877	2.99
BU7832	Straight Talk About Computers	business	1389	19.99
BU2075	You Can Combat Computer Stress	business	0736	2.99

Figure 2-7: A join

The Transact-SQL **select** statement that returns the highlighted data from the two tables is:

```
select *
    from publishers, titles
    where publishers.pub_id = titles.pub_id
    and city = “Berkeley
```

Other Commands

You can do more with Transact-SQL than just query. Transact-SQL includes other commands that create tables and views, such as **create table** and **create view**. It also includes commands that modify tables (the **insert**, **update**, and **delete** commands), and commands that perform many other database tasks discussed in this manual.

Transact-SQL offers extensions to the standard library of SQL commands. We have already looked at one powerful extension, in “Stored Procedure” on page 2-5. Another is **control-of-flow language**, Transact-SQL’s programming-like constructs that control the flow of execution of the Transact-SQL statements. When combined with Transact-SQL's sophisticated error handling techniques, including the ability to capture a **return status** and get reports from **global variables**, they make Adaptive Server a fully programmable database server.

The Transact-SQL commands are described in the *Adaptive Server Reference Manual*. For in-depth discussions of how to use many of the Transact-SQL commands, see the *Transact-SQL User's Guide*.

Presenting Transact-SQL to the RDBMS

There are several ways to present Transact-SQL commands to the RDBMS. One way is to use Sybase's interactive query parser, **isql**. You can invoke **isql** at the operating system command line and then enter Transact-SQL commands at the **isql** prompt. You can also use SQL Advantage, which is a GUI-based query parser, accessible from the Sybase program group or the Utilities section of Sybase Central on Windows or Windows NT clients.

End users typically do not use SQL itself but access the databases using specialized client applications. The client applications send Transact-SQL commands to the RDBMS by passing them as parameters to Open Client library routines (see "Building or Buying the Client Applications" on page 3-7).

System Database

There are two kinds of databases: **system databases** and **user databases**. User databases are databases that you create to store and manage your enterprise data. System databases are databases that the system uses to manage the system.

The *master* database is a system database that is installed with every Adaptive Server. It stores information about all the user databases and other system databases and their associated devices.

The *sybsystemprocs* database is the database that stores all the system procedures provided by Sybase.

The *model* database is a system database that Adaptive Server uses as a template for creating user databases. You can customize the default *model* database that Sybase provides to include your own database objects. For example, if you have created some special stored procedures that you want included in all the user databases that are subsequently created, you can accomplish this by creating the procedures in the *model* database.

The *tempdb* database is a system database that Adaptive Server uses to provide temporary working storage for the objects it creates in the process of executing queries.

The *sybsecurity* database is an optional system database that you can install if you want to use Adaptive Server's audit system. For information about the audit system and the *sybsecurity* database, see the *Security Administration Guide*.

The *sybsyntax* system database is an optional system database that stores the syntax for all the Transact-SQL commands, Sybase system procedures, Adaptive Server utilities, and some Open Client routines. Although the syntax for these facilities is available in the Sybase documentation, if you want to be able to access the syntax quickly online, using the **sp_syntax** system procedure, you must install the *sybsyntax* database.

For more information about the system databases, see the *System Administration Guide*.

Optional Sample Databases

The *pubs2* and *pubs3* databases are optional sample user databases that are used in most of the examples appearing in the Adaptive Server documentation. If you want to try some of the examples that appear in the Adaptive Server documentation, install the *pubs2* or *pubs3* database as described in *Installing Adaptive Server and OmniConnect on UNIX Platforms*. These databases are described in the *Transact SQL User's Guide*.

Database Device

A **database device** stores Adaptive Server databases. It can be a disk or a portion of a disk or, sometimes, an operating system file.

The system and user databases reside on database devices connected to the server. The *master* database must be installed on a device called the master device, which is automatically created at install time. The other system databases and the user databases should be created on other devices.

Part of the process of installing Adaptive Server consists of preparing the database devices. This procedure is explained in detail in *Installing Adaptive Server and OmniConnect on UNIX Platforms*.

System Table

Every user database contains a set of **system tables**, special tables used by the system to manage data and the system. The *master* database contains some special system tables, which do not appear in other databases, that keep track of Adaptive Server as a whole. In other relational database products, what Sybase calls the system tables may be called the data dictionary or the system catalogs.

Adaptive Server Enterprise includes a poster illustrating the contents of and relationships among the system tables. The *Adaptive Server Reference Manual* also describes each system table in detail.

Although you can query the system tables directly to retrieve information about the system, such as the names of all the user databases or all the user accounts on the server, it is usually simpler to use the appropriate system procedure. The system procedures are designed to retrieve the information that a user or System Administrator is likely to need, and they save you the trouble of having to formulate complex queries and become familiar with the details of the system tables on your own.

Permission

A **permission**, sometimes called a privilege, is the authority to perform an action on a certain database object. Adaptive Server permissions are similar to permissions at the operating system level, but Adaptive Server maintains its own permissions for each user, apart from those conferred by the operating system. Examples of Adaptive Server permissions are permission to select data from a table, to modify data in a table, to execute a particular stored procedure, or to create databases or tables.

An Adaptive Server System Administrator or the owner of a particular database is responsible for setting up and maintaining permissions for various users, or groups of users, for Adaptive Server or for a particular database.

Utility Programs

Adaptive Server utility programs perform RDBMS tasks. You invoke them from the operating system rather than through **isql** or a similar program.

isql is a utility. Another utility is **bcp**, which copies bulk data from an operating system file to a database table and viceversa. See *Utility Programs* UNIX Platforms for information on the Sybase utilities.

Transaction Log

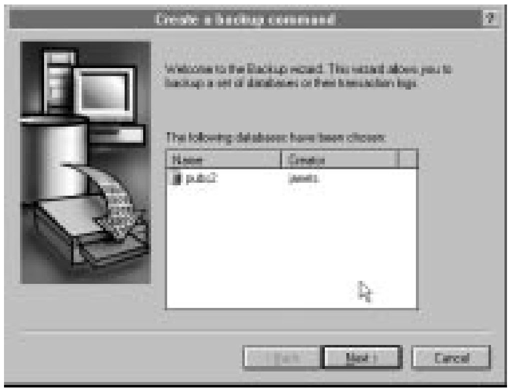
The **transaction log** is a special system table that Adaptive Server maintains for each database to record modifications to the data in the database tables. If a system failure occurs while data is being modified, Adaptive Server uses the transaction log to recover the data and restore the database to the state it was in before the failure occurred. Although the log is part of the database, it should be stored on a separate device to, reduce the likelihood of a physical disk error damaging both a database and its associated transaction log.

System Administration

The term **system administration** refers to an assortment of tasks including, but not limited to, managing Adaptive Server's physical storage, creating and backing up databases, creating user accounts, granting permissions, and running diagnostic and repair functions.

The user responsible for system administration is the Adaptive Server **System Administrator**, who may or may not be the same individual or individuals who are system administrators for the server machine's operating system. For Adaptive Server, the System Administrator can log into Adaptive Server as the special, predefined user "**sa**", using the System Administrator password. The "sa" account has special privileges not held by normal users. The System Administrator can grant other users permission to perform certain system administration tasks.

Sybase Central allows you to execute the most common system administration tasks using wizards. A wizard is a special kind of dialog box or linked dialog boxes that guide you through a task. Figure 2-8 shows a Sybase Central wizard.


Create a backup command

Welcome to the Backup wizard. This wizard allows you to backup a set of databases or their transaction logs.

The following databases have been chosen:

Name	Creator
pubs2	[illegible]

Figure 2-8: A Sybase Central wizard

Some tasks must be executed using Transact-SQL commands or Adaptive Server utilities directly. See Sybase Central's online help for information about administration tasks you can perform from Sybase Central. See the *System Administration Guide* and the *Performance and Tuning Guide* for complete information about Adaptive Server system administration.

3 Using Adaptive Server Enterprise

This chapter briefly describes common tasks that a user performs with a new Adaptive Server. Its gives general information about how to begin using this product.

This chapter covers the following topics:

For complete technical information about Adaptive Server, see the online documentation in SyBooks.

Installing Adaptive Server

You need to install Adaptive Server and the other server components on the server machine, and the client components on one or more client machines. You can install some of the client components on the server, if you find that more convenient. On both servers and clients, you install the software into a Sybase installation directory that you can designate. You should also install SyBooks in order to have access to the online documentation.

If you have an earlier release of Adaptive Server already installed, you may need to upgrade your existing databases before you can access them with the newly installed release.

To install the software, follow the instructions in *Installing Adaptive Server and OmniConnect on UNIX Platforms* and Release Bulletin, which are packaged with the software media.

Configuring Adaptive Server

Configuration of Adaptive Server falls into two categories: post-installation configuration and maintenance configuration.

When you install Adaptive Server, the installation program sets configuration parameters to default values. After installing, you need to configure Adaptive Server for your setup. This involves:

- Setting up client / server network connections
- Specifying the location of the **error log**
- Localizing the language, **character set**, and **sort order**
- Setting certain operating system values
- Setting the **System Administrator** ("sa") password
- Setting up **auditing**, if desired

See *Configuring Adaptive Server for UNIX Platforms* for detailed information about performing these post-installation configuration tasks.

Maintenance configuration is concerned more with fine-tuning memory and disk storage. See the *System Administration Guide* for information about ongoing Adaptive Server configuration.

To configure Adaptive Server from a Windows client with a graphical interface, use the Sybase Central configuration utility to set configuration parameters and configure network connections. Choose Configure from the Server pop-up menu.

Those who prefer a command line interface can set configuration parameters by invoking the **sp_configure** system procedure from **isql**. **sp_configure** is fully documented in the *System Administration Guide* and the *Adaptive Server Reference Manual*.

Designing Databases

There are two stages of database design: **logical design** and **physical design.** In logical design, you define the entities that will be represented by tables in the database, choose the attributes of those entities (which will be represented by columns in the table), and determine how the entities are interrelated through the designation of primary and foreign keys.

Logical design can be accomplished independently of the particular relational database vendor that your organization chooses to supply its software. Logical design should rely heavily on input from the intended users of the databases, and not just on database and software experts. Ideally, you will have completed the logical design of your user databases before you purchase and install Adaptive Server.

In the physical design stage, you map the logical design to the Transact-SQL data definition commands that will create the databases on the server. These are commands like **create database**, **create table**, **create view, create rule**, and so on.

Creating Databases

The output of the physical design effort is a collection of Transact-SQL commands that create databases and create database objects within the database. To create the databases, execute these **create** commands on the server.

If you use Sybase Central to design your databases, Sybase Central itself sends the **create** commands to the server.

It is also possible, of course, to create databases at the **isql** command line using the Transact-SQL **create** commands. Since most databases consist of several tables, if you use the command line method, it is a good idea to save the original data definition commands in a script file that can be run to re-create consistent images of the database as many times as necessary.

Loading Data

After the database tables have been created, you can fill them with data. You may have to insert the data manually, using either the Transact-SQL **insert** command or a client application that has been written to input the data. In many cases, however, the data already exists either in another vendor's database system or in **operating system files**.

If the data is in a file, or can be converted to a file, you can copy it into a database table using the **bcp** bulk copy utility. See *Adaptive Server Utility Programs* for your platform for information on how to use **bcp**.

If the data is in a Sybase database from an earlier release, the original tables may be accessible after they have been upgraded. See the description of the upgrade process in the installation guide to determine whether the databases can be upgraded.

Creating Indexes

If you are going to load a large amount of initial data into a table after creating it, it is best to create the table indexes after you load the data, because indexes slow performance in commands that insert data. Also, for every row added to the table, a row must also be added to the index. However, well-chosen indexes greatly improve performance on commands that retrieve data, so it is advisable to create indexes on the tables that are frequently queried.

When you create an index on a table, you have to decide on which table column or combination of columns the index will be based. You also have to decide what type of index to create and whether it should be unique. See **create index** in the *Adaptive Server Reference Manual* and the chapter on indexes in the *Performance and Tuning Guide* to learn about selecting useful indexes.

You can create indexes using the Sybase Central Index Creation wizard or at the **isql** command line using the Transact-SQL **create index** command.

Backing Up Data

After you input data and create indexes on the tables, you should back up the databases so that you can restore them in the event of a system failure. You should continue to back up data at regular intervals. How often you back up a database depends on how frequently the data is updated and how costly it would be for you to lose it.

Databases that are constantly being modified need to be backed up frequently; databases containing relatively static data can be backed up less frequently. You should regularly back up production databases, but you can usually be less diligent about backing up databases used for test and development purposes if you can afford to lose their most recent data.

If you experience a **media failure**, you can restore the data from the most recent backups, using Adaptive Server's **restore** command.

You do not have to shut down Adaptive Server to back up or restore a database. Backup Server makes it possible to perform online backup and restore operations while Adaptive Server is running.

You can back up and restore databases and their transaction logs by using the Sybase Central Backup and Restore wizards or by using the Transact-SQL **dump** and **load** commands or at the **isql** command line.

See the *System Administration Guide* for information about developing a backup plan and about backing up and restoring user and system databases.

Assigning Logins, Users, and Permissions

Adaptive Server security is based on a three-tiered system that separates server access, database access, and data access.

To connect to a server, a user must have a **login** on the server. A login does not automatically give the user permission to access data or create database objects, just to connect with Adaptive Server. It consists of a login name and a password.

To give others access to a server, the Sybase System Administrator creates logins using either the **sp_addlogin** system procedure from **isql** or the Sybase Central Add Login wizard.

To enable a user to connect to a database, the System Administrator gives the user a database **user ID**. That database username may or may not be the same as the user's login name. The System Administrator or the **Database Owner** (the user who created the database or was given ownership by the database creator) create database users. You can create users with either the **sp_adduser** system procedure from **isql** or the Sybase Central Add User wizard.

In addition to individual users, the System Administrator or Database Owner can create a **group** of database users in Adaptive Server. The group is identified by a group name and provides a convenient way to grant and revoke permissions to more than one user in a single statement. For example, you can define a "managers" group or a "payroll" group, assign users to the group, and assign the same privileges to all the users in the group by assigning the privileges to the group as a whole. There is also a system-defined group, named "all," to which all database users belong.

Having a login and username gives an individual access to a server and a database, but it does not give automatic permission to select data from tables or views, modify data in tables, or execute stored procedures. To give a user permission to access a database object, the System Administrator or the Database Owner must explicitly grant the user, or a group in which the user is a member, permission to access that object. The System Administrator or Database Owner can grant and deny various permissions to other users to use specific database objects by either using the Transact-SQL **grant** and **revoke** commands from **isql** or using Sybase Central.

Querying and Modifying Data

Most activity between a user and Adaptive Server involves either querying data or modifying data. Querying data is discussed in "Queries" on page 2-7.

Modifications

Modifying data means:

- Adding a new row to a table (the **insert** command or copying in data with the **bcp** utility),
- Deleting a row from a table (the **delete** command), or
- Changing the value of one or more columns in a row (the **update** command).

Transactions

If you are executing multiple data modification commands on the same or related data, you can execute them together in a single **transaction**. A transaction consists of multiple Transact-SQL commands, enclosed between begin and end delimiters, that are executed as an **atomic** command rather than as separate commands. The tables affected by the transaction are locked so that other users cannot make changes that affect the work being accomplished during the transaction. The transaction as a whole is either entirely committed or not committed, ensuring that if the system fails in the middle of an operation, the database is not left in an inconsistent state.

The classic example of a transaction is one that removes money from your savings account (**delete**) and adds it to your checking account (**insert**). You would want to perform the **delete** and the **insert** in a single transaction, so that if the system failed after the **delete** but before the **insert**, the record of the funds would not be lost. If such a failure occurred, the transaction would be rolled back by Adaptive Server's recovery program and the database would appear as though the **delete** had never been executed.

Stored Procedures

If you find that there are commands, or sets of commands, that users will perform frequently, you should consider creating a **stored**

procedure. Even for a single command, a stored procedure usually executes faster because the stored procedure stores the command in a partially processed form. Also, you can control access more securely by giving users permission to execute the stored procedure without giving them permission to access all the objects referenced by the commands in the stored procedure. See the *Transact-SQL User's Guide* for more information on creating stored procedures.

Building or Buying the Client Applications

You have a number of options for establishing a client interface through which Adaptive Server users can access data. You can purchase one of many third-party applications, or you can build your own.

If your organization is very small, and your information needs are simple, you may be able to manage by using a set of Transact-SQL statements and creating some stored procedures. Users could perform queries through an **isql** or SQL Advantage client and direct the results of a query to a file.

If your needs are more complex, and you want to provide a customized interface to users who may not know anything about relational databases or Transact-SQL, you can purchase a client application that has been designed for a particular purpose and runs against an Adaptive Server database from a third-party software company or a VAR (Value Added Reseller). Look for the "Industry Solutions" page in the "Partners and Solutions" section of the Sybase Web site for up-to-date news and reference information about industry-specific applications that run against Adaptive Server.

You can purchase (separately) a development tool, such as PowerBuilder, which creates robust standalone GUI client applications. PowerBuilder takes advantage of the extra functionality of any extended attributes you may have defined in Power-Designer.

After building an application, you deploy it on the client, where it is linked with the Open Client libraries. Typically, application-building tools have their own simplified programming APIs and a graphical interface for program development. Figure 3-1 shows the PowerBuilder development window.

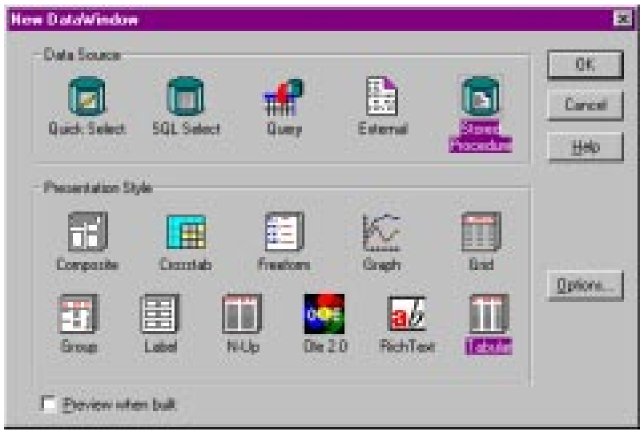


Figure 3-1: PowerBuilder development window

You can also design client applications from scratch using a traditional procedural programming language, such as C, C++, or COBOL. This method usually requires more time and more skilled software engineers than are needed to develop an application using a development tool.

The advantage of the second approach is that the applications are generally smaller, and sometimes faster, because the executable does not have the overhead of the tool. These programs make direct calls to the Open Client library routines to communicate with Adaptive Server. See the Open Client documentation in the Open Client / Server collection in SyBooks for complete information about coding with the Open Client libraries.

Monitoring and Tuning Performance

After building a production database, you may want to examine some performance statistics and adjust some performance parameters to tune the system for high **performance**.

The easiest way to monitor Adaptive Server performance is with Adaptive Server Monitor Server. Monitor Server gives specific details about performance with numeric measurements of cache usage, network traffic, **device I/O**, and **locking** activity, all of which are discussed in the *Performance and Tuning Guide*. You view statistics collected by Monitor Server using Monitor Viewer, which is available from Sybase Central.

Most of your tuning efforts ought to address the amount of time it takes for Adaptive Server to respond to queries. You can achieve high performance by starting out with a well-designed database that includes strategic indexes and by learning to work with the Adaptive Server query **optimizer**, an Adaptive Server feature that analyzes queries and database objects and selects the appropriate query plan, based on how much time it will take to execute. For more information, see the *Performance and Tuning Guide*.

Glossary

Adaptive Server Enterprise

The server in Sybase's client / server architecture. Adaptive Server Enterprise manages multiple databases and multiple users, keeps track of the actual location of data on disks, maintains mapping of logical data description to physical data storage, and maintains data and procedure caches in memory.

APIs

Application Program Interfaces, subroutines that allow client applications to interface with Adaptive Server Enterprise. Also known as **libraries**.

aggregate function

A function that works on a set of cells to produce a single answer or a set of answers, one for each subset of cells. The aggregate functions available in Transact-SQL are: average (**avg**), maximum (**max**), minimum (**min**), sum (**sum**), and count of the number of items (**count**).

application-building tool

See **development tool**.

application server

The programmable part of NetImpact Dynamo that functions like a CGI program and that serves as an interface between a **web server** and Adaptive Server Enterprise.

argument

A value supplied to a function or procedure that is required to evaluate the function.

auditing

Recording security-related system activity that can be used to detect penetration of the system and misuse of system resources.

automatic recovery

A process that runs every time Adaptive Server is stopped and restarted. The process ensures that all transactions that completed before the server went down are brought forward and that all incomplete transactions are rolled back.

backup

A copy of a database or transaction log used to recover from a **media failure**.

business rules

The rules to which the data in a database must conform.

cache

See **data cache**.

character set

A set of specific (usually standardized) characters with an encoding scheme that uniquely defines each character. ASCII and ISO 8859-1 (Latin 1) are two common character sets.

client

The user's side of a client / server arrangement; can refer to the software that is making the calls to the server or to the machine that is running the client software.

client-side script

In a World Wide Web application, a script interpreted by a web client and embedded directly in an HTML page. Compare to **server-side script**.

client/server architecture

A computer system architecture in which clients request a service and a server provides that service. Each machine can specialize in the tasks it is best suited for.

clustered index

An **index** in which the physical order and the logical (indexed) order is the same. A table can have only one clustered index.

column

A data value that describes one characteristic of an **entity**. Also called a **field**.

command

An instruction that specifies an operation to be performed by the computer. Each command or SQL statement begins with a keyword, such as insert, that names the basic operation to be performed.

consolidated database

A database that serves as the “master” database in a hierarchical data **replication** setup. The consolidated database contains all the data to be replicated; while its remote databases can contain only their own subsets of the data. In case of conflict or discrepancy, the consolidated database is considered to have the primary copy of all data.

constraint

A rule applied to a database object that ensures that all entries in the database object to which it applies satisfy a particular condition. For example, a column may have a constraint requiring that all values in the column be unique.

control-of-flow language

Transact-SQL’s programming-like constructs (such as **if**, **else**, **while**, **goto** label) that control the flow of execution of Transact SQL statements.

data cache

Also called named cache or cache. A cache is an area of memory within Adaptive Server that contains the in-memory images of database **pages** and the data structures required to manage the pages.

data definition

The process of setting up databases and creating database objects such as tables, indexes, rules, defaults, procedures, triggers, and views.

data dictionary

The system tables that contain descriptions of the **database objects** and how they are structured.

data integrity

The correctness and completeness of data within a database.

data modification

Adding, deleting, or changing information in the database with the **insert**, **delete**, and **update** commands.

data retrieval

Requesting data from the database and receiving the results. Also called a **query**.

database

A set of related data tables and other database objects that are organized and presented to serve a specific purpose.

database administration

The tasks involved in maintaining, designing, implementing changes to, tuning, and expanding a database. See also **system administration**.

database device

See **device**.

database object

One of the components of a database: **table**, **view**, **index**, **stored procedure**, **trigger**, **column**, **default**, or **rule**.

Database Owner

The user who creates a database. A Database Owner has control over all the database objects in that database. The login name for the Database Owner.

datatype

Specifies what kind of information each column will hold, and how the data will be stored. Datatypes include *char*, *int*, *money*, and so on. Users can construct their own datatypes, based on the Adaptive Server system datatypes.

dbo

In a user's own database, Adaptive Server recognizes the user as "dbo." A **Database Owner** logs into Adaptive Server using his or her assigned login name and password.

decision support

Data used to help managers make business decisions. Usually involves a lot of queries, as opposed to updates.

default

The option chosen by the system when no other option is specified.

default database

The database that a user connects with when he or she logs in.

development tool

Software such as PowerBuilder that helps you build specialized GUI applications for accessing Adaptive Server Enterprise databases.

device

Any piece of disk (such as a partition) or a file in the file system that you specially prepare for storing databases and their objects.

device I/O

Reads or writes to or from a database device.

disk initialization

The process of preparing a database device or a file for Adaptive Server use. Once the device is initialized, it can be used for storing databases and database objects. The command used to initialize a database device is **disk init**.

DLLs

Dynamic Link Libraries. Software used by Microsoft Windows and IBM OS/2 to provide services to applications.

dump

To make a backup of an entire database, including both the data and the **transaction log**, which you accomplish with the **dump database** command. Also, the data that results from this action.

entity

The basic unit described by tables in a relational database. Identifying them is the first step in the **logical design**.

error handling

Techniques available to Transact-SQL programmers for basing code on and displaying errors and error messages.

error log

A file that stores severe error messages and results about the start-up and recovery of databases on a server being started up.

error message

A message issued by Adaptive Server, usually to the user's terminal, when it detects an error condition.

field

A data value that describes one characteristic of an **entity**. Also called a **column**.

foreign key

A key column in a table that logically depends on a **primary key** column in another table. Also, a column (or combination of columns) whose values are required to match a primary key in some other table.

global variable

System-defined variables that are updated by Adaptive Server on an ongoing basis. For example, *@@error* contains the last error number generated by the system.

group

Uniquely named set of users assigned a set of permissions for the objects and operations within a database.

guest

Default user in the *model* database that allows any user with a valid Adaptive Server login to use that database, with limited privileges.

HTML

HyperText Markup Language. The language with which World Wide Web documents are formatted. It defines basic details of the document, such as paragraph tags, graphics, and hypertext links.

HTTP

HyperText Transfer Protocol. The protocol that negotiates document delivery to a **web client** from a **web server**. When a web client requests an **HTML** document from a web server, the server obeys the request, transfers the document, and closes the connection.

index

A database object that consists of key values from the data tables and pointers to the pages that contain those values. Indexes speed up access to data rows by pointing Adaptive Server Enterprise to the location of a table column's data on disk.

integrity rules

Rules that describe how data will be kept accurately and consistently in the relational model.

intranet

A private network that uses Internet software and standards. It allows you to use a web server to publish information within the organization served by the private network.

ISAPI

Internet Server API. A web server programming interface developed by Microsoft.

JavaScript

A cross-platform World Wide Web scripting language. You can embed client-side JavaScript scripts into NetImpact Dynamo templates to allow the web client to handle some processing tasks or to call Java applets.

join

A basic operation in a relational system that links the rows in two or more tables by comparing the values in specified columns.

key

A field used to identify a record, often used as the **index** field for a table.

key value

Any value that is indexed.

keyword

A word or phrase that is reserved for exclusive use by Transact-SQL. Also known as a reserved word.

libraries

See **APIs**.

load

To restore data stored in a backup created during a **dump**.

local variables

User-defined variables defined with a **declare** statement.

locking

The process of restricting access to resources in a multiuser environment to maintain security and prevent concurrent access problems. Adaptive Server automatically applies locks to tables or **pages**.

logical design

Defining the tables, relations, and keys of a relational database. Distinguished from **physical design**.

logical key

The primary, foreign, or common **key** definitions in a database design that define the relationship between tables in the database. Logical keys are not necessarily the same as **physical key**s (the keys used to create indexes) on the table.

logical read

The process of accessing a data page or an index page that is already in memory, to satisfy a query. Compare to **physical read**.

login

The name a user uses to log into Adaptive Server. A login is valid if Adaptive Server has an entry for the user in the system table *syslogins*.

MAPI

Microsoft's Message Application Programming Interface, a message system used in several popular e-mail systems such as Microsoft Mail.

Master database

Controls the user databases and the operation of Adaptive Server Enterprise as a whole. Known as *master*, it keeps track of such things as user accounts, ongoing processes, and system error messages.

master table

A table that contains data on which data in another table logically depends. The detail table typically has a foreign key that joins to the primary key of the master table.

media failure

A media failure occurs when the information on a medium (typically a hard disk drive) becomes unusable.

messages

Message-based communication between applications or computers does not require a direct connection. Instead, a message agent sent at one time by an application can be received at another time by another application.

message system

In SQL Remote™ replication, a protocol for exchanging messages between the consolidated database and a remote databases. Usually, a consolidated database and a remote database send and receive messages using the same messaging system.

message number

The number that uniquely identifies an error message.

model database

A template for new user databases. The installation process creates *model* when Adaptive Server is installed. Each time the **create database** command is issued, Adaptive Server makes a copy of *model* and extends it to the size requested, if necessary.

nested queries

select statements that contain one or more subqueries.

nonclustered index

An **index** that stores key values and pointers to data.

normalization rules

The standard rules of database design in a relational database management system.

NSAPI

Netscape Server API. A programming specification for Netscape's web servers to access back-end applications.

null

Having no explicitly assigned value. NULL is not equivalent to zero or to blank. A value of NULL is not considered to be greater than, less than, or equivalent to any other value, including another NULL value.

object permissions

Permissions that regulate the use of certain commands (data modification commands, plus **select**, **truncate table** and **execute**) to specific tables, views or columns.

objects

See **database object**.

ODBC

The Open Database Connectivity (ODBC) interface, a standard interface to database management systems in the Windows and Windows NT environments.

ODBC data source

A database that can be accessed through the **ODBC** interface and for which the appropriate drivers have been installed on the local client. In order to connect to the database, you configure the connection information in the desktop application called the ODBC Administrator.

operating system

A group of programs that translates your commands to the computer, so that you can perform such tasks as creating files, running programs, and printing documents.

operating system file

Collection of data named and recognized by the operating system. Adaptive Server Enterprise data is not stored in so-called operating system files, but it can be exported to operating system files with the Sybase utility, **bcp**.

optimizer

Adaptive Server Enterprise code that analyzes queries and database objects and selects the appropriate query plan. It estimates the cost of each permutation of table accesses in terms of how much processing and disk writing time it will take.

page

A 2K block of data that is the minimal unit that can be read from or written to disk.

parameter

An **argument** to a **stored procedure**.

PDM

Physical Data Model. The PowerDesigner model that specifies a physical implementation of a database design and that is represented in a graphic format. You can create a PDM from scratch or by reverse-engineering an existing database.

PERL

Practical Extraction and Reporting Language. An interpreted scripting language, typically used in writing CGI scripts.

performance

The speed with which Adaptive Server processes queries and returns results. Performance is affected by several factors.

permission

The authority to perform certain actions on certain database objects or to run certain commands.

physical design

Mapping the **logical design** to the Transact-SQL data definition commands that create databases on the server.

physical key

A column name, or set of column names, used in a **create index** statement to define an index on a table. Physical keys on a table are not necessarily the same as the **logical keys**.

physical read

A disk I/O to access a data, index, or log page. Adaptive Server Enterprise estimates physical reads and logical reads when optimizing queries. See **logical read**.

primary key

The column or columns whose values uniquely identify a row in a table.

privilege

The authority to perform certain actions on certain database objects or to run certain commands. Synonymous with **permission**.

projection

One of the basic query operations in a relational system. A projection is a subset of the columns in a table.

query

1. A request for the retrieval of data with a **select** statement.

2. Any SQL statement that manipulates data.

query plan

The ordered set of steps required to carry out a query, complete with the access methods chosen for each table.

relational database management system

A system for storing and retrieving data from two-dimensional tables in which **SQL** use is standard.

recovery

The process of rebuilding one or more databases from database dumps and log dumps.

referential integrity

The rules governing data consistency, specifically the relationships between the primary keys and foreign keys of different tables. Adaptive Server addresses referential integrity with user-defined **triggers** and with referential integrity **constraints**.

relational database

A collection of tables that stored interrelated data.

relationship

Describes how entities are related. A basic step in the logical design of a database is to identify the relationships between the entities you have identified.

remote database

In SQL Remote replication, a database that exchanges replication messages with a consolidated database. Remote databases may contain all or some of the data in the consolidated database.

remote procedure calls

A **stored procedure** executed on a different Adaptive Server Enterprise from the server the user is logged into.

replication

For databases, a process by which the changes to data in one database (including creation, updating, and deletion of records) are also applied to the corresponding records in other database.

restriction

A subset of the rows in a table. Also called a **selection**, it is one of the basic query operations in a relational system.

return status

A value that indicates that the procedure completed successfully or indicates the reason for failure.

roles

Provide individual accountability for users performing system administration and security-related tasks in Adaptive Server. The **System Administrator**, System Security Officer, and Operator roles can be granted to individual server login accounts.

rollback transaction

A Transact-SQL statement used with a user-defined **transaction** (before a **commit transaction** has been received) that cancels the transaction and undoes any changes that were made to the database.

row

A set of related **columns** that describes a specific entity. Also called a record.

rule

A specification that controls what data may be entered in a particular column, or in a column of a particular user-defined datatype.

sa

The login name for an Adaptive Server **System Administrator**.

savepoint

A marker that the user puts inside a user-defined **transaction**. The user can later use the **rollback transaction** command with the savepoint name to cancel any commands up to the savepoint, or **commit transaction** to actually complete the commands.

selection

A subset of the rows in a table. Also called a **restriction**, it is one of the basic query operations in a relational system.

server-side script

In a World Wide Web application, a script interpreted on the WWW server before being sent to the web client. Compare to **client-side script**.

server user ID

The ID number by which a user is known to Adaptive Server Enterprise.

sort order

Used by Adaptive Server to determine the order in which to sort character data.

SQL

Structured Query Language (SQL), the language used to communicate with a relational database and that is the subject of standards set by several standards bodies.

SQL Remote

An asynchronous message-based replication system for two-way server-to-laptop, server-to-desktop, and server-to-server replication.

statement

A statement begins with a **keyword** that names the basic operation or command to be performed.

store-and-forward

An exchange of information, typical of message-based systems, that allows information to be exchanged without a direct connection between applications.

stored procedure

A collection of SQL statements and optional control-of-flow statements stored under a name. Adaptive Server-supplied stored procedures are called **system procedures**.

subquery

A **select** statement that is nested inside another **select**, **insert**, **update** or **delete** statement, or inside another subquery.

system administration

An assortment of tasks including but not limited to managing Adaptive Server's physical storage, creating and backing up databases, creating user accounts, granting permissions, and running diagnostic and repair functions. See also **database administration**.

System Administrator

A user who is authorized to handle Adaptive Server system administration, including creating user accounts, assigning permissions, and creating new databases.

system databases

The databases on a newly installed Adaptive Server: *master*, which controls user databases and the operation of the Adaptive Server; *tempdb*, used for temporary tables; *model*, used as a template to create new user databases; and *sybsystemprocs*, which stores the system procedures. See also **user databases**.

system procedures

Stored procedures that Adaptive Server supplies for use in system administration. These procedures are provided as shortcuts for retrieving information from the system tables or mechanisms for accomplishing database administration and other tasks that involve updating system tables.

system table

One of the data dictionary tables. The system tables keep track of information about Adaptive Server Enterprise as a whole and about each user database. The *master* database contains some system tables that are not in user databases.

table

A collection of **rows** (records) that have associated **columns** (fields). The logical equivalent of a database file.

temporary database

The database in Adaptive Server Enterprise called *tempdb* that provides a storage area for temporary tables and other temporary working storage needs.

throughput

The volume of work completed in a given time period. It is usually measured in transactions per second.

Transact-SQL

The SQL dialect used in Adaptive Server Enterprise.

transaction

A grouping of a series of Transact-SQL statements so that they are treated as a single unit of work. Either all statements in the group are executed or no statements are executed. The tables queried during the transaction are locked until the transaction is complete.

transaction log

A **system table** (*syslogs*) in which all changes to the database are recorded.

trigger

A special form of **stored procedure** that goes into effect when a user gives a change command such as **insert**, **delete**, or **update** to a specified table or column. Triggers are often used to enforce referential integrity.

unique indexes

Indexes that do not permit any two rows in the specified columns to have the same value. Adaptive Server checks for duplicate values when you create the index (if data already exists) and each time data is added.

URL

Uniform Resource Locator. A standardized character string that identifies the location of a World Wide Web document.

update

An addition, deletion, or change to data, involving the **insert**, **delete**, **truncate table**, or **update** statements.

user database

A database created by a user. See also **system databases**.

user-defined datatype

A definition of the type of data a column can contain that is created by the user. These **datatypes** are defined in terms of the existing system datatypes. Rules and defaults can be bound to user-defined datatypes (but not to system datatypes).

user ID

The ID number by which a user is known in a specific database. Distinct from **server user ID**.

variable

An entity that is assigned a value. Adaptive Server has two kinds of variables, **local variables** and **global variables**.

view

A named **select** statement that is stored in the database as an object. It allows users to “view” a subset of rows or columns from one or more tables.

web client

Also known as a web browser. Client software that requests and displays **HTML** documents and other Internet or **intranet** resources.

web server

A computer that serves data over the World Wide Web. Also, an application that receives and serves data in the **HTTP** protocol over the Internet or a corporate **intranet** and turns a computer into a web server. Sometimes called an HTTP server or an HTTP daemon.

Index