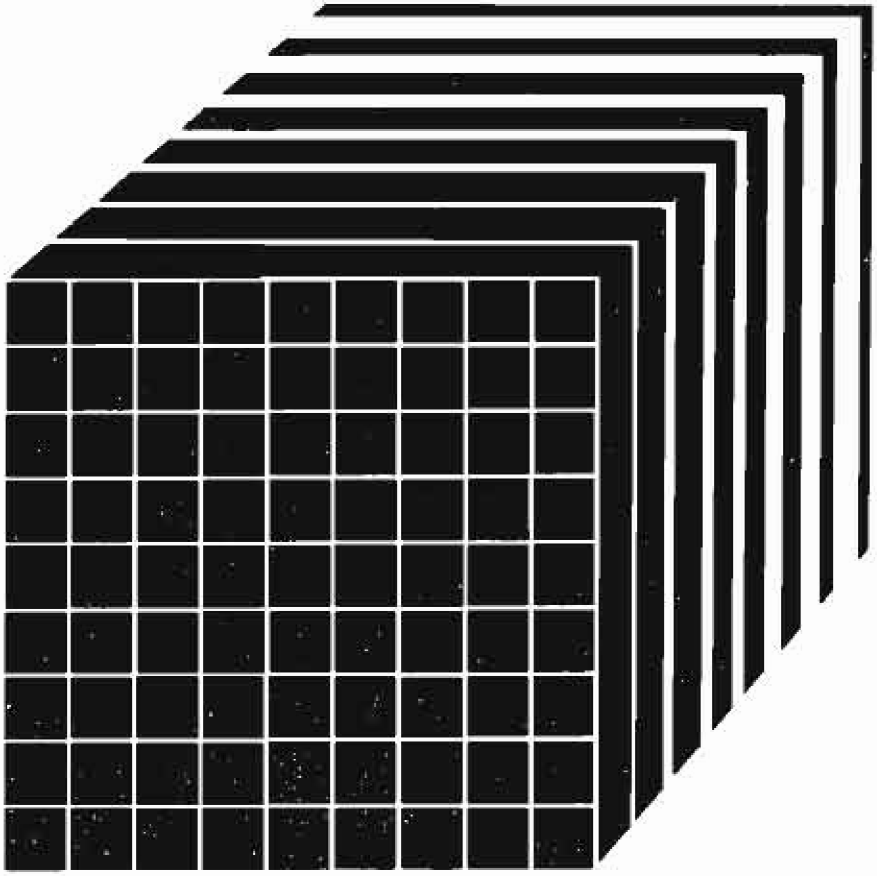
IBM

SQL/Data System General Information for VM/System Product

Release 3.5

GH24-5064-1



Second Edition (November 1985)

This edition, GH24-5064-1, is a revision of GH24-5064-0. This edition applies to the Structured Query Language/Data System (SQL/DS, Program Number 5748-XXJ) in a VM/System Product (VM/SP) environment. This edition applies until otherwise indicated in new editions or Technical Newsletters. Changes are periodically made to the information herein. Before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370 and 4300 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

Summary of Changes

For a detailed list of changes, see page iii.

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

In this manual are illustrations in which names are used. These names are fanciful and fictitious; they are used solely for illustrative purposes and not for identification of any person or company.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Ordering Publications

Requests IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. Publications are *not* stocked at the addresses given below.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to:

IBM Corporation
Department 6R1T
180 Kost Road
Mechanicsburg, Pennsylvania
U.S.A. 17055

or:

IBM Canada Ltd.
Information Development
Department 849
1150 Eglinton Avenue East
Don Mills, Ontario
Canada M3C 1H7

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Summary of Changes

SQL/DS Release 3.5

SQL/DS Release 3.5 improves SQL/DS in the following areas:

- Faster archive and restore operations, including a new capability to archive only the log of changes to the data base. In addition, archives of the data base can now be created using non-SQL/DS facilities.

- Greater availability of the data base. If portions of a data base become inoperative, the other portions can still be used.

- Additional aids for finding and fixing problems.

The new facilities are either transparent or used only when requested. You do not need to change your operating procedures if you do not use the new facilities.

For more technical information, refer to *SQL/Data System Concepts and Facilities for VM/SP*, GH24-5065 and the other SQL/DS manuals updated for SQL/DS Release 3.5.

SQL/DS Library of Publications

Technical enhancements have *not* affected all of the SQL/DS books. Only the following SQL/DS for VM/SP books have been updated:

- *General Information for VM/SP*, GH24-5064
- *Concepts and Facilities for VM/SP*, GH24-5065
- *Licensed Program Specifications for VM/SP*, GH24-5066
- *Planning and Administration for VM/SP*, SH24-5043
- *Installation for VM/SP*, SH24-5044
- *Messages and Codes for VM/SP*, SH24-5070
- *Operation for VM/SP*, SH24-5071
- *Diagnosis Guide for VM/SP*, SY24-5230
- *Diagnosis Reference for VM/SP*, SY24-5232
- *Master Index for VM/SP*, SH24-5072

Minor technical and editorial changes have also been included in these manuals.

SQL/DS Release 3

This is a list of technical changes for Release 3 of SQL/DS that affected this manual.

SQL/DS Accounting Facility

This facility keeps track of the amounts of system resources used by each SQL/DS user and SQL/DS process. Installations can "charge" users for the resources they consumed in the SQL/DS virtual machine.

DBS Utility Commit Point/Restart

Users can tell SQL/DS to periodically commit changes to the data base while using the DBS utility to load data into a data base.

Performance Improvements

- Specifying the Isolation Level
  ISQL users, programmers, and DBS utility users can specify whether or not other users can update data that the program has finished reading in the current logical unit of work.

- FETCH and INSERT Blocking
  Programmers can specify that a program retrieve and insert rows in groups. In addition, ISQL uses Fetch Blocking for all queries, and the DBS utility automatically exploits both Fetch and Insert Blocking.

Enhancements for National Languages: Mixing DBCS and EBCDIC Data

SQL/DS can interpret identifiers and character strings that contain both DBCS and EBCDIC data.

SQL/DS Library of Publications

- Independent library for VM/SP users

  For Release 3, two separate libraries of SQL/DS manuals became available: one for VSE users and one for VM/SP users.

- New Diagnosis Manuals

  Two new manuals were added to the library:

  - *SQL/Data System Diagnosis Guide for VM/SP*, SY24-5230
  - *SQL/Data System Diagnosis Reference for VM/SP*, SY24-5232.

  These manuals help in diagnosing problems in SQL/DS. They replaced the *SQL/Data System Logic* manuals.

- Minor Technical and Editorial Changes

Preface

This book is written for customer executives who are evaluating the Structured Query Language/Data System (SQL/DS) for a VM/SP system and deciding if it meets their needs. This book is also intended for anyone interested in an overview of SQL/DS in a VM/SP system; for example, a data base administrator, system programmer, operator, or general user.

For more details about the concepts of SQL/DS and the facilities SQL/DS offers, refer to *SQL/Data System Concepts and Facilities for VM/SP*, GH24-5065.

A complete list of the SQL/DS manuals for VM/SP installations is given under "Documentation" on page 41. Other manuals with general information about products related to SQL/DS are:

IBM Virtual Machine/System Product General Information, GC20-1838

IBM Virtual Machine/System Product: CMS Primer, SC24-5236

IBM DATABASE 2 (DB2) General Information, GC26-4073

Query Management Facility: General Information, GC26-4701.

| *VMBACKUP Management System General Information*, GH20-6248.

Note: If you are interested in SQL/DS on Virtual Storage Extended (VSE) or VSE/System Package (VSE/SP) systems, refer to the IBM publication *SQL/Data System General Information for VSE*, GH24-5012.

Contents

Figures

Highlights

General Description

The Structured Query Language/Data System (SQL/DS) is a relational data base management system designed for both end users and data processing professionals. With SQL/DS, any user can access data in interactive and noninteractive systems.

SQL/DS simplifies the task of handling data. It is designed to be easy to install and use. SQL/DS provides facilities for querying data, manipulating data, and writing reports. It also contains data recovery and data security measures.

The language for handling data is the Structured Query Language (SQL). SQL commands can be entered from a display terminal, used as input to the utility program, or embedded in a program. Programs can be coded in COBOL, PL/I, FORTRAN, or assembler language.

SQL/DS presents all data as tables. All data appears in row and column format. This sample table has three rows and four columns:

```
CHECK    DATE         RECIPIENT            AMOUNT
-----    ----         ---------            ------
  101    10/3/84      MORTGAGE COMPANY     262.00
  102    10/6/84      AUTO REPAIR           40.50
  103    10/9/84      TAX BUREAU           478.35
```

Using the Data Base

SQL/DS provides data base capability that is easy to use and easy to administer. With little effort, a data base can be set up and end users can begin to use SQL/DS almost immediately. In a development center environment, SQL/DS provides data base capability for interactive, dynamic use and supports changing requirements. It is a useful tool for modeling data designs and developing uses of data.

Querying Data

With many programming and data languages, you must code a long sequence of instructions explaining how to get data, what to look for, and what to do with it. In SQL/DS, you can do all this with one command. Even better, you don't have to know how or where the data is stored. You just have to know the names of the table and its columns.

SQL/DS lets users access data directly, using SQL commands, without the need for a user-written program. Several users can access SQL/DS data at the same time. They can query and update data or create reports while other users and programs are working with other data in the same or different tables. SQL/DS has facilities which resolve conflicting updates.

SQL commands can do many common data processing tasks, such as:

- Select (and perhaps update) some item or group of items.
- Sort data.
- Calculate values using the data.

Putting such work in SQL commands instead of in a program uses the power of SQL/DS and decreases the data handling done by programs. A few SQL commands executed from a terminal can sometimes replace a program.

Writing Reports

Users can store query commands along with format commands. Format commands control how query results are displayed or printed. A user can develop the format of a report at a terminal and then request a printed copy of the query result seen at the terminal.

This query and report writing capability can replace some of the need for special programs. Thus, SQL/DS users can have access to current data quickly.

Using Application Programs

SQL/DS data can be accessed from programs and terminals. Programs work with the data through SQL commands embedded in them. The detailed task of accessing the data is done by SQL/DS so programs are less affected by changes in data format. Programs that access SQL/DS data can also access other data, such as CMS or VSAM files.

Data Recovery

SQL/DS can recover data if there is an abnormal ending of a program, a media failure, or a system failure. The recovery facilities include dynamic backout, logging, and archiving.

Archiving (copying data onto magnetic tape) can involve the entire data base, or just the log of data base activity. Restoring involves copying the entire data base from tape and then applying the log archives (if any) and the log.

SQL/DS can perform the complete archiving and restoring operations. However, with the user archive/restore capability, other programs, such as the VMBACKUP Management System or the DASD Dump Restore (DDR) facility, may be used.

SQL/DS also permits recovery using portions of the log and allows sections of the data base to be disabled for user access. The recovery process is controlled by the administrator of the data base.

Data Security

SQL/DS also enforces security measures. SQL/DS controls access to SQL/DS and to data managed by SQL/DS. SQL/DS has a security audit facility that records who is using SQL/DS and the access privileges and authorities for each user.

Data Administration

Data administration means deciding what data is to be stored in the system and controlling the use of that data. In SQL/DS, users can do much of their own data administration. The activity permitted to each SQL/DS user is controlled via the security authorization facilities of SQL/DS. For example, depending on the privileges they have, users can design, create, alter, and delete their own data.

A group of users can also be put in charge of the data base system. These **data base administrators** control the system resources (for example the data base itself, the users' data, etc.). In SQL/DS, there can be one data administrator or many. SQL/DS does not require that one person be appointed data administrator or a new group be set up to control SQL/DS. Your installation can decide how much centralized data control it needs.

SQL/DS has catalogs which contain information on the data in the data base, and each user's authority to access parts of that data. These catalogs, which are part of SQL/DS, serve as an integrated data dictionary and directory. They are dynamically updated to reflect the current status of the data base. When a user defines a data base object (creates a table, for example), SQL/DS catalogs are automatically updated. The catalogs make it easier to keep track of data stored in the system.

SQL/DS is normally run so that many users and programs can access SQL/DS data at the same time. This is called multiple user mode. SQL/DS can also run in single user mode when maximum performance on involved, time-consuming activities is required. Such activities include loading large quantities of data or working on complex queries that access large amounts of data.

Productivity Increases

The primary aim of SQL/DS is to improve the productivity of people who work with data. SQL/DS:

- Requires minimum planning.
- Requires minimum skills.
- Reduces program maintenance caused by changes in data arrangement.

The productivity effect of SQL/DS on various groups is as follows.

End Users

Users of SQL/DS (especially those who use just the query and report writing facilities) need not have data processing training. The concepts of tables and their uses are common ones, especially for people accustomed to keeping records.

The simplicity of the SQL/DS data concept reduces the time and cost for education. New users can learn about SQL/DS from the tutorial *SQL/Data System Terminal User's Guide for VM/SP*. Later they can use the *SQL/Data System Terminal User's Reference for VM/SP* manual. Or they can display the online reference (HELP) information as they work at the terminals.

The time needed to formulate and enter a query at the terminal is much less than the total time needed to code and execute an equivalent one-time programming request. Thus more timely information can be used in decision making.

Application Programmers

Programmers can use SQL/DS interactively to satisfy one-time programming requests *without* writing a program.

Programmers who design or write programs for SQL/DS need not be concerned with physical data organization. SQL/DS simplifies the handling of data. SQL/DS removes from the programmer the job of physically handling the data records. Therefore, programmers are free to work on the logic of the programs. This can improve programmer productivity and reduce the program backlog.

Programmers can use SQL/DS interactively to develop and test SQL commands for later use in programs. Online testing can increase programmer productivity by reducing the total time for program development.

SQL and the data table concept reduce the burden on the programmer. Because SQL is easy to use, the probability of SQL coding errors is low. If an error is made, it is easy to detect and correct.

When it is time to revise the data base, programmers often don't have to make extensive changes to their programs. Data restructuring may result in little or no change to programs. Therefore, the data processing staff can respond easily to changing business needs.

Administrators

Many administrative decisions can be deferred until users have gained some experience with an initial set of data tables. SQL/DS default parameters can be used for initial operations. Later, after the system is better known and its usage is established, the system can be adjusted to meet specific goals. Adjustments can be made while the data is in use. Only major changes require that SQL/DS be halted.

Computer Operators

SQL/DS requires a minimum of data processing expertise to support it. Installing SQL/DS is straightforward. Operations scheduling is easy because definition or change of one part of the data can be done while users are working with other parts. There is little or no time lost for data reorganization.

Some Basics of SQL

SQL is easy to use, not only for programmers, but also for personnel who are not involved with data processing. A person no longer has to learn programming to store and maintain records on a computer system.

SQL lets people work with data without their having to learn a complex storage format. SQL uses the same language structure for commands entered at the terminal interactively, as for commands coded in a program.

A main use of SQL is in querying stored data. SQL is also used for defining data, manipulating data, and controlling access to data.

Note: *Only basic aspects of SQL are shown in this book. For more about SQL, refer to SQL/Data System Concepts and Facilities for VM/SP or the tutorial SQL/Data System Terminal User's Guide for VM/SP.*

Traditionally, much of the data used in business, engineering, and science is organized in tables. As a result, the SQL/DS data format using rows and columns is very familiar. An example SQL/DS table is shown in Figure 1.

Table: EMPLOYEES

SERIAL	NAME	DEPARTMENT	TELEPHONE
79	HENRY, G. G.	ACCOUNTING	2111
101	BOSWELL, T. U.	ACCOUNTING	2211
126	ANDERSON, R. T.	SALES	6312
5846	ANDREWS, B. K.	ACCOUNTING	2112
2312	BAKER, C. J.	SHIPPING	3756
4684	DENNIS, W. G.	RECEIVING	3855
6587	JOHNSON, A. V.	RECEIVING	4422

Figure 1. A Table Named EMPLOYEES

Query Commands

A query command (SELECT) entered from a terminal yields the entire result in the display.

A query can contain logical operations (such as "greater than") or calculations (including built-in functions like maximum or average). A query can also contain a join operation. A join operation allows a user to select data from more than one table at a time. This join capability is described in *SQL/Data System Concepts and Facilities for VM/SP.*

The basic form of a query is:

SELECT some data (names of columns)

FROM some place (names of tables)

WHERE some conditions exist (if any)

Here are some examples of simple queries:

Problem: Display the entire table named EMPLOYEES.

Solution:

```
SELECT  *  FROM  EMPLOYEES
```

yields:

```
SERIAL  NAME                   DEPARTMENT   TELEPHONE
------  ----                   ----------   ---------
    79  HENRY, G. G.           ACCOUNTING        2111
   101  BOSWELL, T. U.         ACCOUNTING        2211
   126  ANDERSON, R. T.        SALES             6312
  5846  ANDREWS, B. K.         ACCOUNTING        2112
  2312  BAKER, C. J.           SHIPPING          3756
  4684  DENNIS, W. G.          RECEIVING         3855
  6587  JOHNSON, A. V.         RECEIVING         4422
```

Note: The asterisk (*) in the command denotes "all columns" of the table.

Problem: List in alphabetic order the Accounting Department employees and their telephone numbers.

Solution:

```
SELECT   NAME, TELEPHONE   FROM  EMPLOYEES
                           WHERE  DEPARTMENT='ACCOUNTING'
                           ORDER BY  NAME
```

yields:

```
NAME                   TELEPHONE
----                   ---------
ANDREWS, B. K.              2112
BOSWELL, T. U.              2211
HENRY, G. G.                2111
```

Problem: List in numeric order the employees whose serial numbers are greater than 5000.

Solution:

```
SELECT   SERIAL, NAME   FROM  EMPLOYEES
                        WHERE  SERIAL > 5000
                        ORDER BY  SERIAL
```

yields:

```
SERIAL   NAME
------   ----
  5846   ANDREWS, B. K.
  6587   JOHNSON, A. V.
```

Data Definition Commands

The basic data definition command is the CREATE TABLE command. The command to create the EMPLOYEES table (Figure 1) is:

```
CREATE TABLE   EMPLOYEES   (SERIAL       INTEGER,
                            NAME         VARCHAR(20),
                            DEPARTMENT   VARCHAR(10),
                            TELEPHONE    INTEGER)
```

The CREATE TABLE command gives the name of the table, the name of each column, and the type of data in each column. In this example, the keyword VARCHAR, used for the NAME and DEPARTMENT columns, means that these columns will hold character strings of varying lengths. The numbers in parentheses tell how many characters can be in an entry in those columns. The CREATE TABLE command does not put any data into the table.

Another command is ALTER TABLE, which lets users add a new column to a table. For example:

```
ALTER TABLE  EMPLOYEES  ADD  SALARY DECIMAL(8,2)
```

defines a new column named SALARY to the EMPLOYEES table. The data in the SALARY column will be decimal numbers of up to 8 digits; 2 of those digits will be to the right of the decimal point. The column now exists, but all values in the new column are null. Null represents a value that is unknown or not applicable. A null value is commonly represented in a display by a question mark (?); however, users can change this representation if they wish. The values will remain null until a user or a program fills in the data by using the UPDATE command. A user who now entered

```
SELECT * FROM EMPLOYEES
```

would see:

```
SERIAL  NAME                DEPARTMENT    TELEPHONE    SALARY
------  ----                ----------    ---------    ------
    79  HENRY, G. G.        ACCOUNTING         2111         ?
   101  BOSWELL, T. U.      ACCOUNTING         2211         ?
   126  ANDERSON, R. T.     SALES              6312         ?
  5846  ANDREWS, B. K.      ACCOUNTING         2112         ?
  2312  BAKER, C. J.        SHIPPING           3756         ?
  4684  DENNIS, W. G.       RECEIVING          3855         ?
  6587  JOHNSON, A. V.      RECEIVING          4422         ?
```

Data Manipulation Commands

Data manipulation commands change existing data or put in new data. These commands are INSERT, DELETE, and UPDATE. Each usage affects one or more rows or parts of one or more rows in a table. For example, when a new employee is added to the Sales Department, a command like:

```
INSERT INTO EMPLOYEES VALUES(5253,'WILLS, J.','SALES',3943,200)
```

places one new row in the EMPLOYEES table:

```
SERIAL  NAME                DEPARTMENT    TELEPHONE   SALARY
------  ----                ----------    ---------   ------
  5253  WILLS, J.           SALES              3943   200.00
    79  HENRY, G. G.        ACCOUNTING         2111        ?
   101  BOSWELL, T. U.      ACCOUNTING         2211        ?
   126  ANDERSON, R. T.     SALES              6312        ?
  5846  ANDREWS, B. K.      ACCOUNTING         2112        ?
  2312  BAKER, C. J.        SHIPPING           3756        ?
  4684  DENNIS, W. G.       RECEIVING          3855        ?
  6587  JOHNSON, A. V.      RECEIVING          4422        ?
```

A single command can affect multiple rows in a table. For example, the command:

```
DELETE FROM  EMPLOYEES  WHERE DEPARTMENT='ACCOUNTING'
```

erases from the EMPLOYEES table all the rows for the Accounting Department employees:

```
SERIAL  NAME                DEPARTMENT    TELEPHONE   SALARY
------  ----                ----------    ---------   ------
  5253  WILLS, J.           SALES              3943   200.00
   126  ANDERSON, R. T.     SALES              6312        ?
  2312  BAKER, C. J.        SHIPPING           3756        ?
  4684  DENNIS, W. G.       RECEIVING          3855        ?
  6587  JOHNSON, A. V.      RECEIVING          4422        ?
```

The following commands update the EMPLOYEES table with some salary data:

```
UPDATE EMPLOYEES  SET SALARY = 220 WHERE SERIAL = 0126
UPDATE EMPLOYEES  SET SALARY = 185 WHERE SERIAL = 2312
UPDATE EMPLOYEES  SET SALARY = 190 WHERE SERIAL = 4684
UPDATE EMPLOYEES  SET SALARY = 195.50 WHERE SERIAL = 6587
```

The result is:

```
SERIAL  NAME                DEPARTMENT    TELEPHONE   SALARY
------  ----                ----------    ---------   ------
  5253  WILLS, J.           SALES              3943   200.00
   126  ANDERSON, R. T.     SALES              6312   220.00
  2312  BAKER, C. J.        SHIPPING           3756   185.00
  4684  DENNIS, W. G.       RECEIVING          3855   190.00
  6587  JOHNSON, A. V.      RECEIVING          4422   195.50
```

A single command can affect only parts of multiple rows in a table. For example, the command:

```
UPDATE  EMPLOYEES  SET SALARY = 1.15 * SALARY
                   WHERE  DEPARTMENT = 'SALES'
```

gives a 15% raise to all employees in the Sales Department:

```
SERIAL  NAME                DEPARTMENT   TELEPHONE   SALARY
------  ----                ----------   ---------   ------
  5253  WILLS, J.           SALES             3943   230.00
   126  ANDERSON, R. T.     SALES             6312   253.00
  2312  BAKER, C. J.        SHIPPING          3756   185.00
  4684  DENNIS, W. G.       RECEIVING         3855   190.00
  6587  JOHNSON, A. V.      RECEIVING         4422   195.50
```

View and Authorization Commands

A CREATE VIEW command allows the creator of a table to define a certain way of looking at some or all the data in that table. A CREATE VIEW command can deal with more than one table.

Authorization commands (GRANT and REVOKE) control the types of authority that users or programs can have.

A simple use of CREATE VIEW and GRANT gives a user the authority to display (but not change) part of a table. For example, the command:

```
CREATE VIEW  SALESV  AS  SELECT  NAME,SERIAL,TELEPHONE
                         FROM    EMPLOYEES
                         WHERE   DEPARTMENT='SALES'
```

defines a view named SALESV. It consists of just three columns from the EMPLOYEES table. The command:

```
GRANT SELECT ON SALESV TO JOHNSON
```

grants to user JOHNSON the privilege of selecting the data in view SALESV. User JOHNSON cannot see other data (such as SALARY) about Sales Department employees. Also, JOHNSON cannot see any data about employees in other departments. Note that the view given to user JOHNSON presents the columns in the order NAME, SERIAL, and TELEPHONE. This is different from the order in the EMPLOYEES table.

User JOHNSON can display all the SALESV data as if it were a table by using a query:

```
SELECT * FROM SALESV
```

with the result:

```
NAME                        SERIAL            TELEPHONE
----                        ------            ---------
WILLS, J.                     5253                 3943
ANDERSON, R. T.                126                 6312
```

The query that user JOHNSON enters is thus simpler than it would be if the query operated directly on the EMPLOYEES table. The query provides just the data that user JOHNSON needs.

Any changes to the main table are reflected in the view. For example, suppose ANDERSON's phone number is changed in the EMPLOYEES table as follows:

```
UPDATE EMPLOYEES SET TELEPHONE = 3434 WHERE SERIAL = 126
```

The next time user JOHNSON displays the SALESV data, it would look like this:

```
NAME                        SERIAL            TELEPHONE
----                        ------            ---------
WILLS, J.                     5253                 3943
ANDERSON, R. T.                126                 3434
```

Using SQL/DS from a Terminal

SQL/DS lets a person work directly with data via an IBM 3270-type (or equivalent) display terminal. By using a communication facility, several users can use the same SQL/DS system. For more information, refer to "Relationships with Other IBM Products" on page 31.

Besides SQL commands, SQL/DS has another set of commands that make working at a terminal easy. These commands are directed to the Interactive SQL Facility (ISQL), which controls the terminal session. ISQL commands are for:

- Formatting Output and Writing Reports

  A user can enter ISQL display control commands while looking at a result at a terminal. For example, a result may have more lines than can fit on a single display screen. To handle this situation, the user can enter ISQL commands to "page" through the result. If a result is wider than the display screen, the user can enter ISQL commands to shift the display left and right as needed.

  ISQL also has format commands to control how results are shown at the terminal and printed in reports. For example, column names can be changed, subtotals and totals can be shown, and report titles can be specified.

- Storing SQL Commands

  SQL/DS lets a user store any SQL command for repeated use. The user defines a name for each stored SQL command, and later uses this name to start each execution.

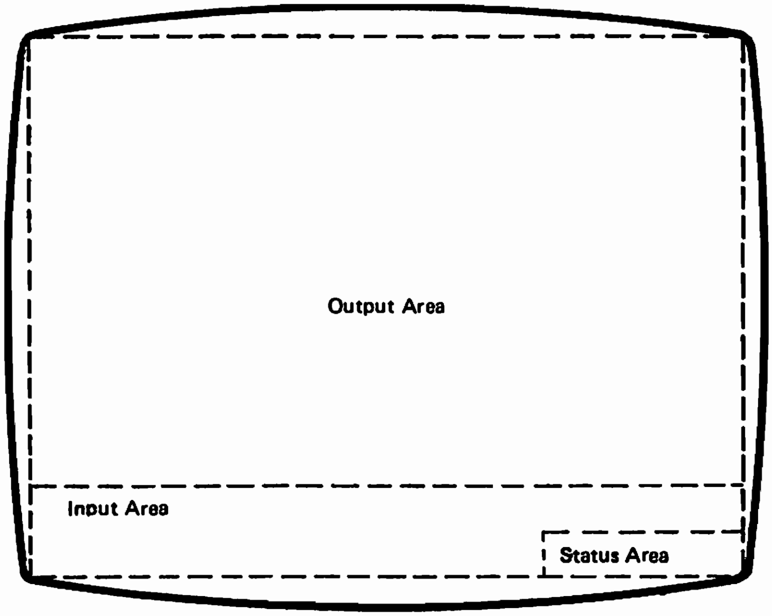
  A user can keep format controls with any stored query so that when that query is executed, the format of the result is as desired. A user can develop the format for a report and then save both the query and the format controls for repeated use.

- Displaying Online Reference (HELP) Information

  HELP information is included when SQL/DS is installed. An SQL/DS user can easily display it as an aid while working at the terminal. This reference information covers many topics, including all SQL/DS messages, SQL commands, and ISQL commands.

Screen Format

ISQL uses an IBM 3270-type (or equivalent) display terminal as a line-oriented input/output system. Query results are displayed as a formatted full screen. Depending on the display terminal used, the user is presented with a 24- to 32-line by 80- to 132-character screen. The system screen format for a 24 X 80 mode terminal is shown in Figure 2.



The input area is where the user keys in command.

The status area is where VM/SP provides system status information.
For example: VM READ

Figure 2. ISQL Screen Format

The input area is two lines, minus the characters used for the status area. Where color display terminals are available, the user can define both color and highlight characteristics for the ISQL screen format.

Program Function Keys

Commonly used ISQL commands are set in the Program Function (PF) keys of the terminal keyboard to reduce the typing needed for various tasks. For example, certain keys carry out the forward, backward, left, and right movements useful in reading lengthy results. If desired, the user can change the PF key settings.

Report Writer

Some of the ISQL commands are for transforming query results into formatted reports. A user can display a result on the screen, adjust its format, and then print the report. Reports can be printed on either a system printer or a work station printer. This portion of ISQL provides for:

- User-defined (or default) titles for the top and bottom of each page.
- Page number and date at the top of each page.
- Subtotals and totals for numeric columns in the report, grouped as specified by the user.
- Punctuation control for decimal data. (Different uses of periods, commas, and spaces can be specified for marking decimal positions and thousands positions.)
- Multiple copies of a report.

Storing SQL Commands

An ISQL user can key in any SQL command and keep it for later use. This saves time for the users because they don't have to key in the entire SQL command each time.

The user specifies the name under which the SQL command is stored. The user can then recall a stored SQL command and use it just as it was stored; or the user can recall a stored SQL command, change it, and then use it. (ISQL commands are used to store, recall, change, and execute the SQL command.)

SQL commands can be stored with places reserved for values that are different each time the commands are executed. For example, suppose the following query (for the EMPLOYEES table in Figure 1) is stored as NAMETELE:

```
SELECT NAME,TELEPHONE FROM EMPLOYEES
                      WHERE DEPARTMENT = '&1'
                      ORDER BY NAME
```

The stored query can then be used many times, with a different value for &1 each time. For example:

```
START NAMETELE (ACCOUNTING)
     or
START NAMETELE (SALES)
```

ISQL recalls the SQL command stored as NAMETELE, replaces the &1 with the parameter in parentheses, and then submits the SQL command for execution.

Creating and Running Routines

Routines are for people who often use sequences of commands at a terminal. Such a sequence might issue SQL commands or SQL stored queries and generate reports. These sequences, called routines, are stored and invoked by a name, much like stored SQL commands. A frequently used sequence of commands can thus be executed easily, saving time for the user. A routine can contain variables (like the &1 in the preceding example) and can be shared among users.

Education

The time needed to become skilled in using SQL/DS from a terminal depends on the proficiency desired. Skill in doing basic tasks can be gained in less than a day. The tutorial *SQL/Data System Terminal User's Guide for VM/SP* is a good way to learn about SQL and ISQL commands.

Using SQL/DS from an Application Program

An application program that uses SQL/DS data is similar to any other application program. The difference is that the commands that use SQL/DS data are SQL commands embedded in the program.

SQL Commands in Application Programs

SQL simplifies programs. To use SQL/DS data, a programmer specifies the desired table and column names along with any qualifying phrases needed to limit the request, just like an ISQL user.

Only minor differences exist in the SQL commands used in the programming languages supported by SQL/DS. For example, differences exist in the way commands are punctuated and the way program variables are declared. These differences stem from the high-level languages themselves and are natural for users of those languages.

Consider the INVENTORY table and programming examples shown in Figure 3. Each example in Figure 3 shows the declaration of variables and a simple query (SELECT) command. The variables to be used by SQL/DS are declared in a special section of declarations. Note that variables are always preceded by a colon (:) when they are used in an embedded SQL command.

The program places the part number of interest into variable ZZ before executing the SELECT command. The SELECT command gets the two fields DESCRIPTION and QUANTITY from the INVENTORY record for a certain PARTNUMBER (ZZ).

Figure 3 shows a SELECT command that returns a single row result. SQL/DS also makes it possible for a user to return a *set* of records, namely a table as a whole. (This is one characteristic of the relational data model.) For a query that returns more than one row in a result, a *cursor* must be used. (If the query is in a FORTRAN program, the cursor must be used, regardless of the number of rows returned in the query result.) A cursor acts like a pointer in the set of rows returned by a query. The program uses the cursor to retrieve the result one row at a time and, if desired, to update or delete each row.

Table: INVENTORY

```
PARTNUMBER DESCRIPTION QUANTITY
---------- ----------- --------
       207 GEAR              75
       209 CAM               50
       221 BOLT             650
       222 BOLT            1250
       231 NUT              700
       232 NUT             1100
       241 WASHER          6000
       285 WHEEL            350
       295 BELT              85
```

PL/I Example:

```
EXEC SQL BEGIN DECLARE SECTION;
    DCL XX CHAR(24);
    DCL YY BIN FIXED(31);
    DCL ZZ BIN FIXED(15);
    ●
EXEC SQL END DECLARE SECTION;
    ●
EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY
 FROM INVENTORY WHERE PARTNUMBER = :ZZ;
    ●
```

COBOL Example:

```
DATA DIVISION.
WORKING-STORAGE SECTION.
   EXEC SQL BEGIN DECLARE SECTION END-EXEC.
77 XX PICTURE X(24).
77 YY PICTURE S9(9) COMPUTATIONAL.
77 ZZ PICTURE S9(9) COMPUTATIONAL.
     ●
   EXEC SQL END DECLARE SECTION END-EXEC.
     ●
PROCEDURE DIVISION.
     ●
   EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY
   FROM INVENTORY WHERE PARTNUMBER = :ZZ END-EXEC.
     ●
```

Figure 3 (Part 1 of 2). Examples of SQL Commands in Programs

Assembler Language Example:

```
   EXEC SQL BEGIN DECLARE SECTION
XX      DS  CL24
YY      DS  F
ZZ      DS  H
        ●
   EXEC SQL END DECLARE SECTION
        ●
   EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY         C
            FROM INVENTORY WHERE PARTNUMBER = :ZZ
        ●
```

FORTRAN Example:

```
 EXEC SQL BEGIN DECLARE SECTION
   CHARACTER*24   XX
   INTEGER        YY, ZZ
      ●
 EXEC SQL END DECLARE SECTION
      ●
 EXEC SQL DECLARE C1 CURSOR FOR
*      SELECT DESCRIPTION, QUANTITY
*      FROM INVENTORY
*      WHERE PARTNUMBER = :ZZ
 EXEC SQL OPEN    C1
 EXEC SQL FETCH   C1 INTO   :XX, :YY
 EXEC SQL CLOSE   C1
      ●
```

Note: PL/I, COBOL, and Assembler language programs can also be coded using the "DECLARE-OPEN-FETCH-CLOSE" format required for the FORTRAN programs.

Figure 3 (Part 2 of 2). Examples of SQL Commands in Programs

Preprocessing the SQL Commands in an Application Program

SQL commands in a program must be interpreted before the program is compiled or assembled. This step is done by submitting the program source code to one of the SQL/DS preprocessors. SQL/DS has one preprocessor for each of the following programming languages — COBOL, FORTRAN, PL/I and assembler language.[1]

A preprocessor produces changed source code for that language's normal compiler/assembler and then saves an access module for that program in the SQL/DS data base. The access module contains code for the SQL

[1] SQL commands can also be embedded in APL2 programs. APL2 provides its own interface to SQL/DS and dynamically executes SQL commands (much like ISQL). SQL/DS, itself, does not provide an APL2 preprocessor.

commands in the program. When the program is later run, it will call that module every time it accesses SQL/DS data.

A preprocessor verifies that the user has the authority to access the SQL/DS data used in the program. A preprocessor also updates the SQL/DS catalogs. This gives the user who preprocessed the program the privilege of using the access module. (That user can then grant this privilege to others.) SQL/DS also includes an Extended Dynamic Command facility. This facility provides for precompiled access to SQL/DS for preprocessing SQL commands. This allows programmers to write their own language preprocessors, application development programs, and other tools.

A programmer also has two optional performance parameters that can be specified for a program at preprocessing time: the Blocking option and the Isolation Level options. The Blocking option is for programs that will run in multiple-user mode. If this option is specified, the program inserts or retrieves SQL/DS rows in groups. This can improve performance.

There are two Isolation Level options that specify whether other users can update data that the program has finished reading. Isolation Level Cursor Stability can improve performance for programs running in multiple-user mode. Isolation Level Repeatable Read offers extra protection against updates for data used by the program.

If these options are chosen carefully for the program during preprocessing, performance will improve when the program is run. ISQL and Data Base Services (DBS) utility users can also specify Isolation Levels. The DBS utility is discussed on page 28.

Programmed Queries in an Application Program

Many query applications arise from the needs of groups of end users. An installation can tailor queries to these needs by using application programs.

Application programs that carry out planned queries use system resources more efficiently than unplanned queries issued from a terminal. The programmed SQL commands are preprocessed only once, but they are run many times. Each program run then does not involve the computations needed to interpret an unplanned query.

Advantages of SQL/DS for Application Development

SQL/DS has the following advantages in each stage of program development.

- SQL commands refer to the data by name. Then SQL/DS finds the data in the data base and returns it in the order specified. A programmer does not need to know how the data is stored.

- Each programmer working on an independent application can easily copy data and maintain separate data for testing.

- Preprocessing a program catches SQL-related errors. Programmers do not have to fully compile and test their programs just to catch SQL syntax errors.

- Initial tuning can be done apart from program testing.

Installing and Using the Application

- New programs and tables can be installed at any time. SQL/DS has no generation requirements for adding new applications or changing old ones.

- New columns can be added to existing tables *without reorganizing* the data or recompiling existing programs. Programs can still access the old columns. Only programs that use the new columns need to be changed.

- SQL/DS performance can be improved at any time by adding indexes as desired. Programs do not need to be changed to take advantage of indexes, but they do need to be preprocessed again. On the other hand, if an index is deleted, only performance (not data access) is affected.

- SQL/DS does not interpret embedded SQL commands at execution time. (That work was done once at preprocessing time.)

- SQL/DS enforces security measures to limit data access to those users or programs specifically authorized.

- Recovery facilities of SQL/DS protect against data loss or damage caused by hardware or software failure.

Planning Period

- End users are familiar with data shown in rows and columns. An application designer does not have to translate computer data storage structures into forms the end user can understand. Thus the planner can work directly with the familiar row and column format and avoid misunderstandings with the end user.

- Planners can use ISQL or the DBS utility to create prototype tables, load them with sample data, and show sample application usage. End users can then see the proposed scheme in operation *before* it is coded. Such a prototype can save both computer time and people time. Design flaws can be more easily corrected at this early phase.

- Planners can create a prototype using production data. End users or programmers can then access the prototype via terminals until the complete program is available. End users can get some results while their application is under development.

Design Phase

- Data design can be straightforward. The designer can design tables without the need to foresee all intended usage of the data. The designer does not have to make an optimal design for all applications. The designer can focus on the data needs of the application.

- SQL/DS provides ways to tune the system *without* affecting program design. Tables can be designed *first* and performance can be tuned *later*.

- Training time for designers and programmers can be brief because of the simplicity of data arranged in tables and the ease of use of SQL commands.

- End users familiar with SQL/DS use the same language (SQL) that programmers use. Therefore, communication between them is clear.

Program Coding and Testing

- Programmers can test SQL commands via terminals before they put them into a program, without affecting existing data.

- Programmers can load tables, compile programs, and test the programs while SQL/DS is in operation. They can use terminals to display data changes made by the programs.

- Multiple users can share test tables. Program development and testing can be done concurrently by different users using a common test data base.

Additional Features

Entering Data into SQL/DS

Data can be entered into SQL/DS by any of these methods:

- Terminal users can insert single or multiple rows of data via the Interactive SQL (ISQL) facility.
- The SQL/DS Data Base Services (DBS) utility program can load data from a sequential file, allowing for intermediate commit points.
- A user-written program can load data from an existing file.

Data Types

SQL/DS supports ten types of data. All data in SQL/DS tables must be one of these types. All data in a column must be of the same type.

A user can store numeric data, character data, or Double-Byte Character Set (DBCS) data into SQL/DS tables.

- Numeric data can be whole numbers, decimal numbers, or double-precision floating point numbers.
- Character data can have a fixed number of characters or a varying number of characters.
- The DBCS data types support character sets that require two bytes of storage for each character in the character set. Kanji is one example of such a character set. DBCS data can also have a fixed number of characters or a varying number of characters.

ISQL users can specify whether characters entered in quotes are to be converted to uppercase or stored as entered. SQL/DS can interpret character strings that contain both DBCS and EBCDIC data.

Data Independence

Having data in SQL/DS tables gives terminal users and programmers the freedom to ignore details concerning physical data storage. Users and programs are thus free of data storage details.

The choice of columns in the tables is made by considering the various demands that might be made on the data. Usually, data independence is greater if each table contains only one kind of data (such as parts data, or employee data). This approach decreases duplication of data. It also makes the data base more flexible.

Each user or program then works with a portion or view of the data in the tables. As new needs arise, new columns may have to be added to the existing tables. But the users and the existing programs not concerned with the new needs are not affected.

Data independence means that users and programmers do not have to spend time changing their methods and programs just because data elements they do not need have been modified. They can work on the new needs of the business.

Data Security

SQL/DS enforces security measures to limit data access to authorized users or programs. Each user is defined to SQL/DS in terms of a user identification (userid), a password, and level of authority. Users must first log on to the VM/SP system with a system identifier (userid) and password. SQL/DS then uses this information to determine proper user access. The user's level of authority determines the user's freedom in using SQL/DS. At a typical installation:

- All users are granted CONNECT authority to access SQL/DS.
- Some users are granted RESOURCE authority to enable them to acquire table space. (They consume some of the system's resources).
- A few users are granted DBA (Data Base Administrator) authority. They can perform administration tasks, such as adding new users or assigning table space to other users.

SQL/DS checks a user's authority when a user attempts an operation requiring RESOURCE or DBA authority.

Users can grant to others the privileges of working with the data or programs that they control. Granting and revoking of privileges is done by using SQL commands. For example, the creator of a table can authorize another user to work with only certain rows or columns of that table. The creator can limit the type of work that others may do on that table; for example, usage can be limited to query (SELECT command) activity.

- Application System (AS), Release 3 or later, supports SQL/DS for retrieval, analysis, and updating of relational data. AS is an interactive application development facility.

- Host Data Base View (HDBV), Release 1 or later, provides an IBM Personal Computer user with access to the facilities of a host VM/SP system running IC/1, QMF, or Data Extract (DXT). It can also download sequential files created by the DBS utility.

Related IBM Program Products

IBM DATABASE 2 (DB2)

SQL/DS and DB2 are both relational data base management systems. They both use SQL, and are broadly compatible. DB2 operates in Multiple Virtual Storage (MVS) environments.

The compatibility of DB2 and SQL/DS permits the use of SQL data base managers in establishments that have multiple systems. For example, an establishment might choose to operate a central MVS system running DB2, with additional VSE or VM/SP systems running SQL/DS.

- Data Interchange – Data can be copied (unloaded) from an SQL/DS data base and loaded into a DB2 data base. The SQL/DS Data Base Services (DBS) utility does the unloading, and the DB2 Load utility does the loading. Conversely, sequential files created from DB2 data base data can be loaded into SQL/DS tables via the DBS utility.

- Program Development and Use – Program development can be done on either the DB2 or SQL/DS system. Later execution of the program can be done on either system (or on both systems). Both DB2 and SQL/DS can be used for "production" or data query and data analysis. The compatibility of the two systems provides the flexibility to choose whichever system is most appropriate for each application (and its development).

Data Extract (DXT)

DXT is a tool for extracting data from hierarchical or sequential files (such as an IMS data base, VSAM files or SAM files).. When DXT is used to extract data from other sources, the output can be passed directly to the SQL/DS DBS utility to load that data into an SQL/DS data base.

Information Management System (IMS)

IMS is a hierarchical data base management system used in MVS and OS/VS1 environments. Because SQL/DS and IMS do not operate in the same system environments, SQL/DS and IMS cannot coexist in the same system. However, SQL/DS can be used with DXT to support multiple systems. For example, IMS data can be extracted (copied) via DXT and loaded into an SQL/DS data base via the DBS utility. That data can then be used for further processing or analysis with application programs or queries entered via ISQL or QMF.

Planning and Administration

Planning and administration tasks associated with SQL/DS are described briefly in this section. Much more information is given in the *SQL/Data System Planning and Administration for VM/SP* manual.

System Design

System design can be seen as physical design, data design, and methods design. Physical design is largely capacity planning. Data design deals with creating the tables. Methods design is planning how the system will be used so that data security, data integrity, and data recovery needs are met. These aspects are all related to one another.

Physical Design

Physical design, largely capacity planning, focuses on questions like:

- How much data will be stored?
- What devices (processor, storage, terminals) will be needed?
- How will the data be distributed across the storage devices?
- How will the data be used? Query activity? Operational activity? Both? What activity will use the most resources?
- How many users will there be? How many of them will use SQL/DS at the same time?

Questions like these should be answered in terms of the initial system and needs in the near future.

When an SQL/DS data base is generated, a service program for SQL/DS configures it according to the parameters specified by the definer. Most of these parameters can be altered after the data base is in use. However, a few parameters limit the capacity of the data base and can be changed only by redefining the data base.

A key element in the physical planning is defining the online storage spaces to be used by SQL/DS. At the start, the SQL/DS data base only contains information about its ultimate limits. These limits can reflect a data base far larger than the disk space actually allocated. Space is actually used only as data enters the system.

Data Design

Designing the formats of the tables entails answering questions such as:

- What data should be in SQL/DS tables?
- How can the data be placed in tables so that it can be used easily? Which tables will be used together?
- What data will be processed by programs? End users? Both?
- Which applications need special consideration?

Questions like these should be answered, at least tentatively, before starting to use SQL/DS. However, SQL/DS is flexible; most of these decisions can be easily changed as time goes on.

Methods Design

Methods design involves planning how users will interact with SQL/DS. Such planning entails dealing with SQL/DS both as a whole and as a collection of separate data tables. Different methods will be needed for different situations. Questions in this area are:

- What procedures will be set up for operating SQL/DS? What types of archiving will be done? How often? Using which programs?
- Which users will access online data? How will they use the data (create, update, delete, select only)? What authority limits should be placed on each user?
- Which programmers will use SQL/DS? Will there be teams of programmers on separate projects? Will they share data? What authority will each have?

As with other aspects of SQL/DS initial answers to questions like these can be changed without major impact to daily operations.

General Administrative Support

Administration of SQL/DS includes establishing system policies, practices, and procedures for recovery and security, and supporting performance tuning and problem determination. In each of these areas, many of the design decisions and activities can be delegated to individual users and application designers. However, some level of administrative assistance would typically be desired.

Installation

SQL/DS is designed to be very easy to install. Installing a basic version of SQL/DS consists of:

1. Installing the SQL/DS code.

2. Generating a starter data base.

IBM provides aids for installing the SQL/DS code and generating the starter data base.

Installation Planning

Note the hardware and software cited under "Requirements of SQL/DS" on page 33.

If a location has an earlier release of SQL/DS installed, it is easy for them to move to Release 3.5. A new SQL/DS starter data base or an existing SQL/DS data base can be used to install the new release. For a location that does not have an earlier release of SQL/DS, IBM provides aids for generating a starter data base.

Installation Aids

To help a location get started with SQL/DS, IBM provides procedures to install and generate a starter data base. This starter data base can initially be used to verify that the system has been installed, and for limited SQL/DS educational activities. It is easy to expand or alter this initial data base later.

SQL/DS has internal default values so that installing and starting the system can be done with minimal effort. SQL/DS can be used with the default values until plans are set for changing them to meet local needs. It is recommended that you use the SQL/DS starter data base allocations and specifications for your initial SQL/DS data base.

Documentation

IBM provides:

- A library of manuals for SQL/DS. Each addresses one or more general tasks associated with SQL/DS.
- Online reference information for display at a terminal.
- Sample data tables and sample programs to assist an installation in getting started with SQL/DS.

SQL/DS Library

The SQL/DS library for VM/SP contains the items listed in Figure 5.

Note: *Not all SQL/DS publications have been updated for Release 3.5. In Figure 5, the * by the order number of a publication indicates that it has been updated for Release 3.5.*

SQL/DS Title	Order Number	Purpose	In BOF
General Information for VM/SP	GH24-5064*	Presents a brief overview of SQL/DS.	YES
Concepts and Facilities for VM/SP	GH24-5065*	Gives a closer look at the principles of SQL and their implementation in SQL/DS. This book provides basic information for data processing personnel evaluating SQL/DS.	YES
Planning and Administration for VM/SP	SH24-5043*	Describes the system programming and data administration aspects of SQL/DS for systems operating in a VM/SP environment.	YES
Installation for VM/SP	SH24-5044*	Gives the step-by-step procedures for installing SQL/DS on a VM/SP system. These steps include generating a data base.	YES

Figure 5 (Part 1 of 2). SQL/DS Library for VM/SP

SQL/DS Title	Order Number	Purpose	In BOF
Terminal User's Guide for VM/SP	SH24-5045	Is a tutorial on basic uses of SQL and ISQL commands in VM/SP. This guide is for all beginning users of SQL/DS (end users and programmers). Readers need not be data processing professionals. The guide includes exercises that can be done at a terminal or on paper.	YES
Terminal User's Reference for VM/SP	SH24-5067	Is for terminal users who have completed the Guide. DBS utility users can also reference this book for information on SQL commands. This reference book documents the SQL and ISQL commands in alphabetic order.	YES
Terminal User's Reference Summary for VM/SP	SX24-5142	Summarizes the formats of the ISQL and SQL commands commonly used at a terminal.	YES
Program Function Key Template for VM/SP	SX24-5143	Fits around the program function keys of an IBM 3270 terminal. It is a memory aid for the ISQL functions set into those keys.	NO
Application Programming for VM/SP	SH24-5068	Is the guide and reference book for programmers using SQL. This book presents a complete description of all SQL commands.	YES
Data Base Services Utility for VM/SP	SH24-5069	Is a guide and reference manual for users of the Data Base Services (DBS) utility program.	YES
Messages and Codes for VM/SP	SH24-5070*	Explains each SQL/DS message and completion code. Each explanation includes the action the receiver is to take when that message or code occurs.	YES
Operation for VM/SP	SH24-5071*	Explains how to control SQL/DS via commands from the system operator's console.	YES
Licensed Program Specifications for VM/SP	GH24-5066*	Defines the warranty criteria for SQL/DS.	NO
Diagnosis Guide for VM/SP	SY24-5230*	Provides information on diagnosing functional and performance problems.	YES
Diagnosis Reference for VM/SP	SY24-5232*	Provides reference information on the internal working of SQL/DS for use in problem diagnosis.	YES
Master Index for VM/SP	SH24-5072*	Consolidates the indexes of the preceding manuals into one so that a reader can find which manuals discuss various topics.	YES

Figure 5 (Part 2 of 2). SQL/DS Library for VM/SP

Bill Of Forms (BOF) Numbers for Ordering Publications

SBOF-3239

In Figure 5, the column "In BOF" indicates if that item is included in Bill Of Forms SBOF-3239, which can be used to simplify ordering of this set of items. Ordering one SBOF-3239 is the same as ordering one of each of the items indicated. This BOF number is now included in the IBM System Library Subscription Service (SLSS), thus providing for automatic distribution of any updates to items in the BOF.

SBOF-3240

An additional BOF number is now available for ordering binders for the SQL/DS publications. Ordering one SBOF-3240 is the same as ordering these items:

- Four 3-ring binders (ST00-1779 for SX24-5129-0).
- A sheet of labels (SX24-5133) for the spine pockets of the binders.

SBOF-3240 is *not included* in the IBM System Library Subscription Service.

Items Not in the BOFs

Two items are *not included* in either Bill Of Forms grouping:

- The *Licensed Program Specifications for VM/SP* is not needed in a working library. (One copy is supplied with the SQL/DS product tape.)
- The *Program Function Key Template for VM/SP* is available only in packages of 10 templates.

Persons wishing to learn SQL should start with the tutorial *Terminal User's Guide for VM/SP*. They can then go on to either the *Terminal User's Reference for VM/SP* or the *Application Programming for VM/SP* manual.

SQL/DS is also available for a VSE system. The SQL/DS library for a VSE system is similar to this library. Refer to *SQL/Data System General Information for VSE*, GH24-5012, for details.

Online Reference (HELP) Information

IBM has copied reference information from two manuals (*Terminal User's Reference for VM/SP* and *Messages and Codes for VM/SP*) and has formatted this information for terminal display via ISQL at SQL/DS terminals. This information (supplied with the SQL/DS code) is provided as SQL/DS tables. It can be changed or supplemented to meet local needs. CMS HELP files are also available to provide online reference information for commonly used SQL/DS EXECs.

Sample Data Tables and Programs

The sample data tables are used in the *Terminal User's Guide for VM/SP* as the basis for all examples and exercises. A student can use that guide while working with SQL/DS at a terminal.

The sample programs are documented in the *Application Programming for VM/SP* manual. SQL/DS has one sample program for each language it supports.

Index

GH24-5064-1

SQL/DS General Information for VM/SP (File No. S370/4300-50) Printed in U.S.A. GH24-5064-1

- IBM Virtual Machine/System Product-Entry (VM/SP-Entry), Release 1 or later

In this manual, the term "VM/SP" is used to refer to any one of these operating systems.

For remote printing via ISQL, one of these products (or an equivalent non-IBM product) is required:

- Remote Spooling Communications Subsystem (RSCS) Networking, Version 1, Release 2 or later

- Remote Spooling Communications Subsystem (RSCS) Networking, Version 2, Release 1 or later

SQL/DS includes preprocessors for application programs coded in COBOL, PL/I, FORTRAN, and Assembler Language. Required levels of compilers are as follows:

- Any levels of COBOL, PL/I or Assembler current with the operating system can be used.

- The level of FORTRAN language required is that supported by the VS FORTRAN compiler with the LANGLVL(77) option specified or defaulted. If character strings longer than 254 characters are needed, VS FORTRAN Release 3 (or equivalent) must be used.

The following optional IBM products can also be used with SQL/DS:

- Query Management Facility (QMF), Release 1 or later, provides full-screen query and report writing capabilities.

- SQL/Edit, Release 1 or later, enables full-screen data entry and editing of SQL/DS data.

- Data Base Edit Facility (DBEDIT), Release 1 or later, enables full-screen data entry and editing of SQL/DS data.

- Relational Design Tool (RDT), Release 1 or later, identifies indexes required to support user-specified SQL workloads.

- APL2, Release 1 or later, is an extension of the APL high-level language. It can be used for coding application programs that use SQL commands.

- Information Center/1 (IC/1), Release 1.1 or later, provides an integrated information center environment.

- VMBACKUP Management System, Release 3.1 or later, handles user archive/restore activity on SQL/DS data bases.

Requirements of SQL/DS

Machine Requirements

SQL/DS requires a processor supported by VM/SP. The processor must support conditional swapping and extended floating point.

Minimum real storage space required for SQL/DS is approximately:

- 600K bytes for preplanned application usage.
- 1300K bytes for application development and unplanned query usage.

Additional real storage is needed for the IBM program products associated with SQL/DS and any other products and application programs in the environment.

All devices supported by VM/SP through the Conversational Monitor System (CMS) and Control Program (CP) facilities are also supported by SQL/DS.

ISQL uses VM/SP to support IBM 3270-type terminals or compatible display devices. SQL/DS also supports any system printers or work station printers supported by VM/SP or RSCS Networking.

SQL/DS requires one tape drive for installation. If either SQL/DS archiving or SQL/DS tape tracing is used, a tape drive is needed. If both archiving and tape tracing are used at the same time, two tape drives are needed. (SQL/DS also supports tracing to CMS files on DASD instead of on tape.)

Programming Requirements and Options

SQL/DS will work with these products and following versions, releases, and modification levels unless otherwise stated.

Note: *Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.*

SQL/DS for VM/SP requires an environment provided by one of the following operating systems, either by itself or as the base of a System Offering package:

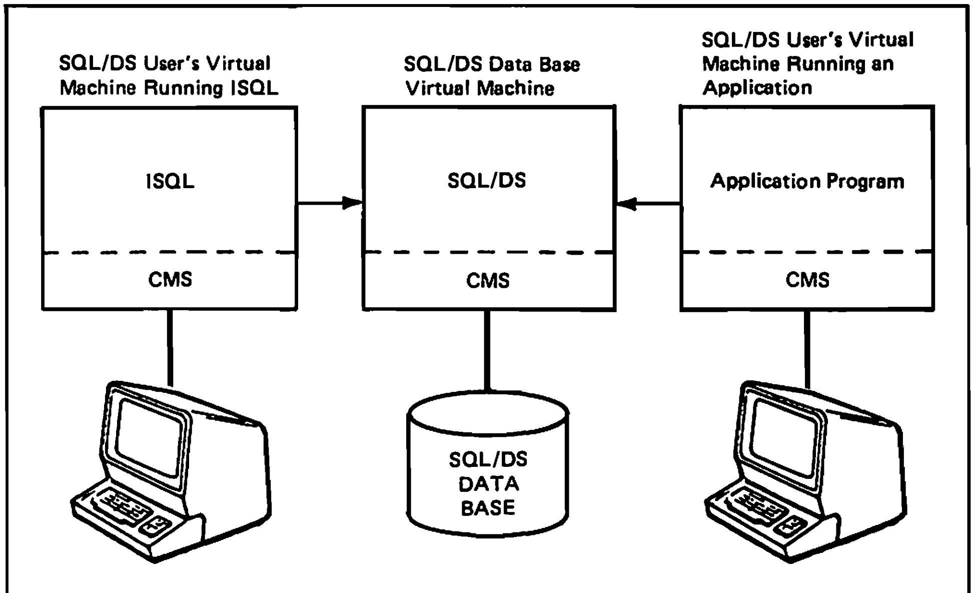
- IBM Virtual Machine/System Product (VM/SP), Release 3 or later
- IBM Virtual Machine/System Product High Performance Option (VM/SP HPO), Release 3.2 or later

Interactive Program Development Environment

SQL/DS works with CMS or equivalent facilities to provide an interactive program development environment. User programs can be developed, preprocessed by an SQL/DS preprocessor, and tested. Also, sequences of commands for the SQL/DS Data Base Services utility may be developed and tested.

Interactive data query and report writing is provided by the Interactive SQL (ISQL) facility. ISQL allows the terminal user to work directly with the SQL/DS data without writing a program. CMS handles the terminal communications while SQL/DS handles the data.

SQL/DS can do interactive and noninteractive work at the same time in multiple-user mode. Figure 4 shows a common arrangement.


SQL/DS User's Virtual Machine Running ISQL
SQL/DS Data Base Virtual Machine
SQL/DS User's Virtual Machine Running an Application
ISQL
SQL/DS
Application Program
CMS
CMS
CMS
SQL/DS DATA BASE

This figure shows two virtual machines communicating with SQL/DS at the same time.

Figure 4. A System Doing Interactive and Noninteractive Work at the Same Time on SQL/DS

Another interactive data query and report writing product that runs with SQL/DS is the Query Management Facility (QMF). For an overview of QMF, refer to the IBM manual *Query Management Facility: General Information*, GC26-4071. Other optional products are listed under "Programming Requirements and Options" on page 33.

Relationships with Other IBM Products

This section briefly describes the system environments in which SQL/DS can work. Some of the IBM products with which SQL/DS works are mentioned here.

Environments

Most of SQL/DS operates in its own virtual machine. It services the data needs of both noninteractive and interactive program development environments.

SQL/DS can operate in multiple-user mode, single-user mode, and multiple data base mode.

- In multiple-user mode, SQL/DS handles data requests sent to it from other virtual machines. A user machine connects to SQL/DS through a link provided by the Inter-User Communication Vehicle facility.

- In single-user mode, the application is in the same virtual machine as SQL/DS. SQL/DS handles the data requests of only that one application.

- In multiple data base mode, several SQL/DS data bases operate at the same time. Different data bases serve then needs of different application areas or user groups. This mode allows for improved accounting and SQL/DS availability. Activity on one data base does not affect user access to the others.

For a more detailed overview, refer to the *SQL/Data System Concepts and Facilities for VM/SP* manual. For complete details, refer to the *SQL/Data System Planning and Administration for VM/SP* manual.

Noninteractive Environment

In a noninteractive environment, SQL/DS provides data handling services to one or more programs operating in virtual machines. This allows production applications to access an SQL/DS data base. The SQL/DS Data Base Services utility program and the SQL/DS preprocessors work in both noninteractive and interactive environments.

automatically when an SQL/DS data base is generated. Users without DBA authority cannot update the catalogs directly (for example, by an UPDATE command).

- Unloads selected data from SQL/DS tables or views to a sequential (SAM) file.

- Reloads (inserts) rows into a table or all tables in a DBSPACE from data that was unloaded into a file by the DBS utility.

- Executes SQL commands against the data base in a noninteractive or interactive mode.

Catalogs

The SQL/DS catalogs are an active dictionary facility. SQL/DS maintains definitions, control information, and general information on data in a set of catalogs. The catalogs are stored as data base tables. They contain "inline" information that controls SQL/DS operation. Besides data definitions, the catalogs contain user descriptions, authorizations, program information, and statistics on the data. They also contain dependency information (such as relationships of views to tables and programs to indexes).

SQL/DS maintains the catalogs automatically, in response to SQL commands that affect them. For example, catalogs are affected when users issue SQL data definition commands (such as CREATE TABLE) or authorization commands (such as GRANT). When one of these commands is entered, SQL/DS updates the catalogs to include the change. This way, a user accessing the catalogs sees live definitions.

SQL/DS validates each definition as it is entered. Incorrect definitions are rejected and a message is displayed to explain the reason.

An important function of the catalogs is change propagation. A change made to one data object often affects other data objects. The catalogs change *all* the affected data objects.

The following are some of the kinds of data kept in the catalogs.

- Data about users — who the SQL/DS users are, what privileges they have, and what data base objects they own.

- Data about programs — the programs that have been defined to SQL/DS and who owns them.

- Data about tables — column names and definitions and which columns are indexed.

- Data about character sets — information for handling national language character sets other than United States English.

Each SQL/DS catalog is a table and can be accessed using SQL commands. Also, because the catalogs are a shared resource, more than one user can access them at the same time. The SQL/DS catalogs are created

- If a certain DBSPACE is known to be in error, that DBSPACE can be taken "off-line." The rest of the data base can then be placed in service. Access to the "off-line" DBSPACE can be restored later.
- If a logical unit of work is known to be in error, the activities done during that unit of work can be bypassed when SQL/DS is restarted.

Accounting

SQL/DS runs as a "service" virtual machine for multiple users. Therefore, VM/SP treats SQL/DS as a single "user" in figuring the system accounting records. All the VM/SP system resources that SQL/DS uses are grouped together into one VM/SP accounting record. An SQL/DS facility must sort out the amounts of system resources used by each SQL/DS user and SQL/DS process.

The SQL/DS Accounting facility keeps track of resources used on SQL/DS. Accounting information is collected for each user, and for each SQL/DS process that cannot be attributed to any one user. The user accounting records reflect each user's demand for SQL/DS. The SQL/DS process accounting records contain information on initialization, checkpoint, and operator processes. Also, at SQL/DS termination time, SQL/DS writes a record showing the total resources used while SQL/DS was in operation. Accounting records are written to the VM/SP system accounting file.

An installation can also code a user "exit." This can be used to supply additional accounting information (such as a project or account number) for the user accounting records.

Data Base Services (DBS) Utility

The SQL/DS Data Base Services (DBS) utility is an SQL/DS application program. It is run as a normal application program, using an SQL/DS-supplied EXEC. It can also be invoked from an application program or from a user EXEC. The DBS utility provides these functions:

- Loads (inserts) data into SQL/DS tables from a sequential (SAM) file, allowing the user to specify periodic commit inserts to the data base. The inserts are saved in the data base every time a specified number of records have been processed. This way, if an error occurs during processing, only the data base changes made since the last commit point need to be reentered.
- Unloads all rows of a table or view or all rows of all tables in a DBSPACE to a specially formatted file.
- Unloads all rows of a table or view or all rows of all tables in a DBSPACE to a sequential (SAM) file.

Each table stored in SQL/DS is placed in some particular portion of the data base called a **DBSPACE**. To store a table in a DBSPACE, the creator of the table must have storing privileges in this DBSPACE. A user who has DBA authority can acquire a DBSPACE for any other user. That user can then create tables in that DBSPACE. Thus, users who have only CONNECT authority can create tables in a limited area; they do not need full RESOURCE authority.

Data Recovery

The SQL/DS recovery facilities include the following:

- When a program ends abnormally, uncommitted data changes will be backed out dynamically. That is, SQL/DS will undo all the changes made to the data since the last time it was committed. Dynamic backout does not interrupt service to other users or programs. Also, applications or users may explicitly back out changes when they detect logical errors.

- If SQL/DS or the system stops abnormally, all uncommitted (incomplete) data changes will be backed out dynamically when SQL/DS is restarted. For this, SQL/DS uses a log kept on a direct access storage device (DASD). The log holds records that describe each change made to the data base. At recovery time, SQL/DS uses the log to restore the data to an earlier state. This restart recovery does not require operator intervention to mount tapes. SQL/DS can be configured with either a single log or dual logs for greater reliability. Logging can also be suppressed.

- Data can be recovered from an archive tape and the log. The archive process places a "backup" copy of the data onto a tape. Archiving is most commonly done when the SQL/DS operator shuts down SQL/DS. Archiving can be done while SQL/DS is serving its users; service to those users is not interrupted. Archiving can involve the entire data base, or just the log of data base activity. Archiving just the log is faster, because the log is smaller than the data base.

  SQL/DS can perform the complete archiving and restoring operations. However, with the user archive/restore capability, other programs, such as the VMBACKUP Management System or the VM/SP DASD Dump Restore (DDR) facility, may be used for data base archives. Under most circumstances, these programs can archive or restore a data base faster than SQL/DS because they take advantage of certain DASD characteristics for high performance.

  SQL/DS also permits extended control of the process of restarting a data base during either a "warm start" or a data base restore operation (for example, after a media failure). If necessary, certain parts of the data base, or certain work that was either committed to the data base or in process can be "filtered out" when SQL/DS is restarted. Here are two examples:

SQL/DS
General Information for VM/SP
Order No. GH24-5064-1

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate. Comments may be written in your own language; English is not required.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

	Yes	*No*
• Does the publication meet your needs?	□	□
• Did you find the material:		
Easy to read and understand?	□	□
Organized for convenient use?	□	□
Complete?	□	□
Well illustrated?	□	□
Written for your technical level?	□	□

- What is your occupation? ______________________________

- How do you use this publication:

As an introduction to the subject?	□	As an instructor in class?	□
For advanced knowledge of the subject?	□	As a student in class?	□
To learn about operating procedures?	□	As a reference manual?	□

Your comments:

If you would like a reply, please supply your name and address on the reverse side of this form.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

GH24-5064-1

Reader's Comment Form

Fold and Tape Please Do Not Staple Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 6R1T
180 Kost Road
Mechanicsburg, Pennsylvania 17055

Fold Fold

If you would like a reply, *please print:*

Your Name ______________________________
Company Name ____________________ *Department* __________
Street Address ______________________________
City ______________________________
State ____________________ *Zip Code* __________
IBM Branch Office serving you ______________________________

IBM®

Cut or Fold Along Line

SQL/DS
General Information for VM/SP
Order No. GH24-5064-1

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate. Comments may be written in your own language; English is not required.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

	Yes	*No*
• Does the publication meet your needs?	□	□
• Did you find the material:		
Easy to read and understand?	□	□
Organized for convenient use?	□	□
Complete?	□	□
Well illustrated?	□	□
Written for your technical level?	□	□

• What is your occupation? ____________________________________

• How do you use this publication:

As an introduction to the subject?	□	As an instructor in class?	□
For advanced knowledge of the subject?	□	As a student in class?	□
To learn about operating procedures?	□	As a reference manual?	□

Your comments:

If you would like a reply, please supply your name and address on the reverse side of this form.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

GH24-5064-1

Reader's Comment Form

Fold and Tape Please Do Not Staple Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 6R1T
180 Kost Road
Mechanicsburg, Pennsylvania 17055

Fold Fold

If you would like a reply, *please print:*

Your Name ____________________
Company Name ____________ *Department* ________
Street Address ____________________
City ____________________
State ____________ *Zip Code* ________
IBM Branch Office serving you ____________________

IBM®

Cut or Fold Along Line