# TWO TIER CLIENT/SERVER DATABASE DEVELOPMENT FOR ALIGNMENT DATA AT THE RELATIVISTIC HEAVY ION COLLIDER AND ALTERNATING GRADIENT SYNCHROTRON*

*F. M. Hemmer, Survey & Alignment Group*
*Brookhaven National Laboratory, Upton, New York 11973-5000 USA*

## 0.1 ABSTRACT

This paper describes the development of the two tier client/server database program, "SurvBase", implemented for accelerator alignment data management at the Alternating Gradient Synchrotron (**AGS**) and the Relativistic Heavy Ion Collider (**RHIC**) at Brookhaven National Laboratory (**BNL**). Integration of Windows NT, Visual Basic 5.0 (**VB5**) and Sybase Open Database Connectivity (**ODBC**) program development environment components is explained. Logic and algorithms for data flow and integrity are given along with Structured Query Language (**SQL**) and VB code examples. Program expansion capacity and back end Relational Database Server switching are discussed.

## 1.0 INTRODUCTION

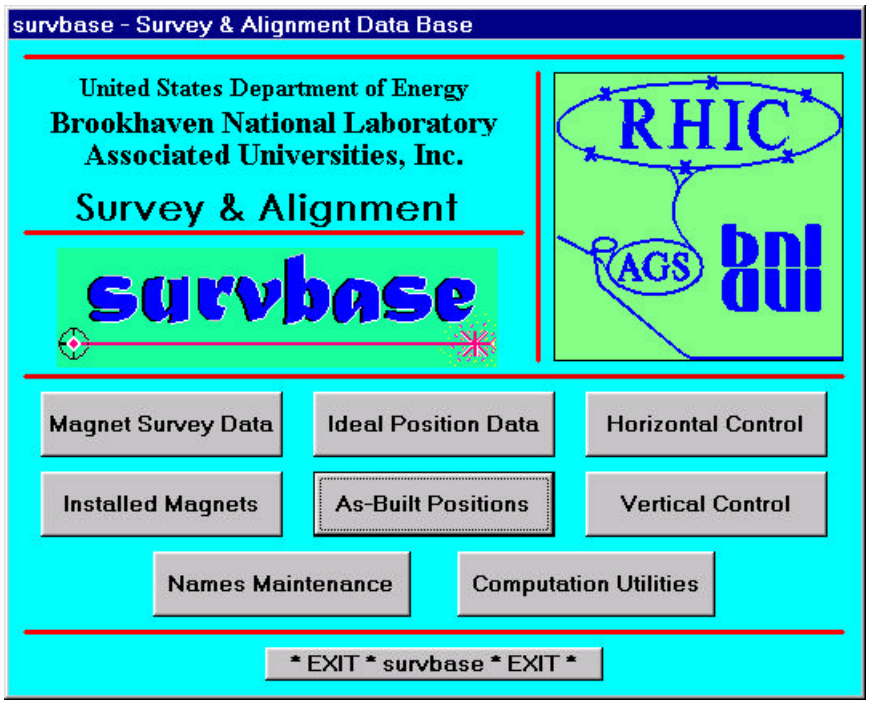


Figure 1. – SurvBase main form.

* Work performed under the auspices of the U.S. Department of Energy

Coordinate data handling for alignment monuments and component fiducials at the AGS/RHIC accelerator complex has been implemented through two tier client server software coded by the author. Figure 1 above shows the opening form of SurvBase. The program SurvBase was originally written in Visual Basic (VB) version 3.0 and made use of MultiLink/VB ODBC drivers from Q+E Software, to enable ODBC interaction with Sybase SQL Server version 4.2. Upon the release of VB version 4.0, Q+E dropped their support for the MultiLink/VB product stating that the functionality it had provided was now built into Visual Basic Professional Edition through the ODBC Application Programmers Interface (**API**) and/or Remote Data Objects (**RDO**).

With the release of VB 5.0, the RDO layer became available only in the Enterprise Edition. The effort to revise the original program code and extend SurvBase with RDO, was begun after the release of VB5 in order to take advantage of very large speed increases resulting from native code compilation available for the first time. VB5 is a 32 bit language and requires the Windows 95 or NT operating system to run its compiled code. SurvBase is under continual development to provide more automated data handling and computation aides.

## 2.0 CLIENT/SERVER

"Client/Server computing involves two or more computers sharing tasks related to a complete application. Ideally, each computer is performing chores appropriate to its design and stated function. This implies that computing resources and data storage resources are located where they will do the most good in fulfilling the computing task at hand." [1] Client/Server describes a program architecture and development process and is not tied to any particular operating system, database engine, programming language or networking environment.

Advantages of client/server applications are task specificity and independence. The tasks that make up the complete application can be identified, segregated and targeted for execution on the most suitable computer. Using ODBC methods and standard programming languages, a complete client/server application can be built so that it can be run without dependence on a particular back end database server, network environment, server hardware or server operating system.

## 3.0 TWO TIER VS. THREE TIER ARCHITECTURE

A two-tier client/server application architecture is implemented when a client talks directly to a server, with no intervening server. It is typically used in small environments of less than 50 users. Generally two-tier architecture separates the user interface and the business logic onto one computer(Tier1) and the database server is onto another computer (Tier2).

In the case of an alignment database, "business logic" is more aptly phrased as "data logic". Data logic is defined as the set of rules and specifications that data or queries must meet before being submitted to the database server. The data logic tier may also include calculation, data analysis or any thing that does not relate specifically to the user interface.

A three-tier client/server application architecture separates the user interface, the data logic and the database server (data services) onto three separate tiers. Generally this means that there are three computers involved, though it is possible to program the application so that the data logic part runs as an independent process on either the client or server computer.

Some database servers have a feature called "stored procedures" allowing the storage and pre-compilation of SQL code in the database itself. Stored procedures execute faster than SQL statements submitted on the fly by a client and are can be used for things such as involved SELECT statements with inner and outer table joins and internal computations. It can be argued that stored procedures in a database represents a restricted form of three-tier architecture. Using this method has the disadvantage of restricting possible changes to the data services tier to those database vendors that support this feature.

Three-tier application development is acknowledged to require more programming time, but in traditional business applications this is weighed against the advantages of the modularization of the code and the ability to attach a different user interface or database server to the most critical business logic tier. Differentiation of program tasks into the appropriate tier, performance oriented coding and detailed planning of the input/output handled by each task, is the foundation of a successful client/server application.

## 4.0 SURVBASE TWO TIER ARCHITECTURE

SurvBase is written as a two-tier application. The user interface, as well as data logic application of rules and formatting to data for submission to the database, are implemented directly in the client side VB5 program code. Some use of stored procedures and the SQL Server ability to return an error for incorrect data or query requests is used to good advantage in the client code. Where certain tasks such as downloading reports or survey name/cogo number synchronization are made available from multiple forms in the user interface, the task code has been placed in globally accessible modules.

## 5.0 SURVBASE DEVELOPMENT

### 5.1 History & Re-Coding

The structure of SurvBase as a two-tier application is due largely to the time available for program development. A significant effort was made in developing the table structure of the underlying database.[2] This basic structure has held up to the test of time with only minor modifications and extensions. Conversely, little effort was made to identify a final form of the SurvBase program in terms of all the data handling and/or computation tasks it would include. Each task was coded separately from beginning to end, including user interface forms, data logic checking and database interaction. During the conversion effort from VB3 to VB5 noted above, some standardization of the user interface forms appearance and re-structuring of program code was accomplished. In addition significant re-coding was required to convert to the RDO method of interacting with SQL Server.

## 5.2 Development Environment

Visual Basic provides an outstanding development environment. Code is presented in different colors according to type and function. Keywords can be automatically completed by the "smart editor". Context sensitive help is always available. During the entry of a function, the correct syntax is presented underneath the cursor as a prompt for function elements. If a variable is dimensioned in the code, the exact spelling including upper/lower case is preserved throughout all modules in the project. When the developer has finished typing a line of code it is evaluated for syntax, conformed to variable spelling and colored. The combined impact of these programming aides improves the developers ability to minimize syntax errors.

Debugging is facilitated while in the design environment by such things as the ability to place the mouse cursor on top of a variable in the code and see its current value without the need for setting a specific variable watch. Break points, step by step execution, next executable statement selection, watch windows and debug statement windows, also enhance the efficiency of the write, compile, run, debugging cycle.

Specific tools for interactive editing of tables and testing of SQL code, such as Interactive SQL Editor (ISQL) which runs natively under NT or Sybase Data Work Bench (DWB) which runs on the server in a Telnet session, are provided by the database vendor. These tools, though somewhat limited in function, serve the specific purpose of testing SQL code interactively, creating and maintaining tables and providing direct access to data.

## 5.3 Performance

Results of the re-coding effort included an increase in the speed of operation of SurvBase, largely due to the RDO method of interaction with SQL Server and the native code compilation capability of VB5.

Using either two or three tier architecture generally implies construction of distributed applications by separating the tiers over a computer network. This means that data base operations have to be performed over the network. In the BNL computing environment the link between the client application and the database server, is over a switched 10mbit Ethernet network. At this data transmission rate the network is not a limiting factor in the perception of speed by the SurvBase end user.

SurvBase performance is good for most operations such as presentation of the current records in a database table or for small batch data loads. There are noticeable lags in response time when processing large data input files where each line individually has queries to the database to check for existence of prior data and/or data rules conformance, followed by the subsequent addition of a new record and/or an update to an existing record in one or more tables. This type of interaction with the database is fairly typical in the alignment data handling process and efforts are now underway to optimize the methods.

### 5.4 Initial Database & Table Setup

The Sybase SQL Server is setup and maintained by the RHIC Accelerator Control Group. Specifics of hardware and software for the both the server and development client are given in Appendix A. A database administrator is responsible for the ongoing health and maintenance of the server. The administrator performs backups and installs upgrades and bug fixes as required. The RHIC database administrator is also available for help with SQL, program performance and table maintenance issues, and has been very helpful during the SurvBase development process.

Creation of the tables for SurvBase was accomplished by writing SQL scripts that were run the ISQL utility creating the tables, columns, indexes, and associated user security permissions. Revisions to tables have been handled by copying the data in a table, dumping the table, revising and running the SQL script and using Batch Control Processing (BCP) to reload the table. This has insured that a complete and current definition of all SurvBase tables is maintained in script form. The following is a sample script that creates the table 'Names' with the columns 'SurveyName', 'cogo', 'regn' and 'ptype', the primary index 'pidx_names' and gives specific permisions to general users (public) and the particular user 'karl'.

```
use survbase
go
create table Names
        (SurveyName varchar(8),
        cogo int,
        regn varchar(8),
        ptype varchar(8) )
go
create unique clustered index pidx_names
        on Names (SurveyName, cogo)
go
grant select on Names to public
go
grant select, insert, update, delete on Names to karl
go
```

## 6.0 DATA LOGIC

The primary goal of the data logic code of a client/server application is to preserve the integrity of the data in the database. Certainly from the accelerator alignment perspective there are serious repercussions associated with using old or mislabeled data. There are two main applications of data logic functions in SurvBase.

The first application is the checking, and formatting of raw coordinate data generated by external computation and analysis software which is to be submitted by the client to the database server. For example, the results generated by a Star*Net least squares control adjustment can be uploaded to either the Horizontal (control) table or the AsBuilt_Horz (fiducials) table. Two Star*Net output files are required to upload the database, the .LST adjustment listing and the .PTS final coordinate points output. After prompting the user for the listing file, SurvBase will check for the existence of the points file. The contents of each file is checked, header

information is parsed and assigned to variables, checks are made for corresponding entries in each file, and temporary arrays are filled with the parsed data. Failure at any step results in notification to the user of the particular problem. Note that up to this point no interaction with the database has been required.

The second application of data logic in SurvBase is the handling of errors returned by the database server when upload data violates internal table integrity or the referential integrity of the database. Syntax errors for submitted queries to the database are also handle by this aspect of data logic.

Referential integrity "insures the integrity of the parent/child relationship created by foreign keys and primary keys." [3] In SurvBase the most common link between tables is the SurveyName column. As shown in the SQL script example in 5.4 above, the Names table is set up with a primary key(index) involving both the 'SurveyName' column and the 'cogo' column. This primary index defines the internal integrity requirements for the Names table and insures a one to one relationship of unique SurveyNames to unique cogo numbers. In other "child" tables the SurveyName column is the primary key but is also a foreign key relating to the Names "parent" table primary key. The effect of this is to insure that only data with valid SurveyName values, already existing in the Names table, can be inserted to the table with the foreign (child) key. The other most common referential integrity relationship in SurvBase involve date/time values.

Referential integrity in Sybase SQL Server is set up by means of foreign key definitions and also trigger definitions. A trigger is "a type of stored procedure often used for enforcing integrity constraints." [4] Triggers are named because of the fact that this particular type of stored procedure is executed (triggered) upon an Insert, Update or Delete event. In SurvBase very little use of triggers has been made since the relationship between tables by common columns and the defined primary/foreign keys suffice to establish referential integrity.

Continuing the Star*Net upload example from above, the SurvBase client sends an ODBC Begin Transaction command to the database server which sets a marker allowing the database to be returned to its exact state prior to any following activity. Subsequently each row in the upload array is extracted, the existence of a known SurveyName in the Names table is checked, the pre-existence of the point with the associated date/time stamp is checked in the target table and the appropriate Insert or Update action is taken. If integrity is violated by the data, SQL Server will return an error that SurvBase must handle. When the complete upload array has been processed successfully, a Commit Transaction command is executed and the database is permanently updated. In the event a data logic error has occurred a Rollback Transaction command is sent to the database and the original state of the database is preserved.

Note that this application of data logic requires interaction with the database and is handled partly by the database server and partly in program code. Parts of the code that would implement the Star*Net upload example can be found in Section 7.2, Example 4 below.

## 7.0 RDO SPECIFIC EXAMPLES

### 7.1 DBGrid & Remote Data Controls

SurvBase makes extensive use of the standard DBGrid and Remote Data Control objects provided by VB5 Enterprise. They are optimized for ODBC and handle data presentation operations quickly. The pairing of a DBGrid control with a Remote Data control provides a quick hook into the back end database for presenting the user with a snapshot of the current table of interest. This same pairing also has the capability of handling simple interactive editing of records in tables, but this has not been used in SurvBase thus far. In general, the database interactions required by SurvBase are for batches of data. The need for single record editing is met by using vendor provided utilities such as ISQL or Data Work Bench.

Figure 2. below shows the AsBuilt Data form in SurvBase. The contents of the form are:

1. Two DBGrid controls which are showing the contents of the AsBuilt_Horz table and the AsBuilt_Vert table
2. Two Remote Data controls (MSRDC1 & MSRDC2 normally hidden) which link to the database server and provide the data displayed by the DBGrid controls.
3. Five Command buttons which will execute the code contained in their _click sub routine when they are clicked by the user.

The DBGrid and Remote Data controls took minutes to draw onto the base form and set up for connection to the database server. The SQL code associated with the two Remote Data controls is simply “SELECT * FROM AsBuilt_Horz” (or AsBuilt_Vert). Scroll bar functioning, column labeling and sizing, etc. are handled automatically by the DBGrid control, but are available for customization by the user.

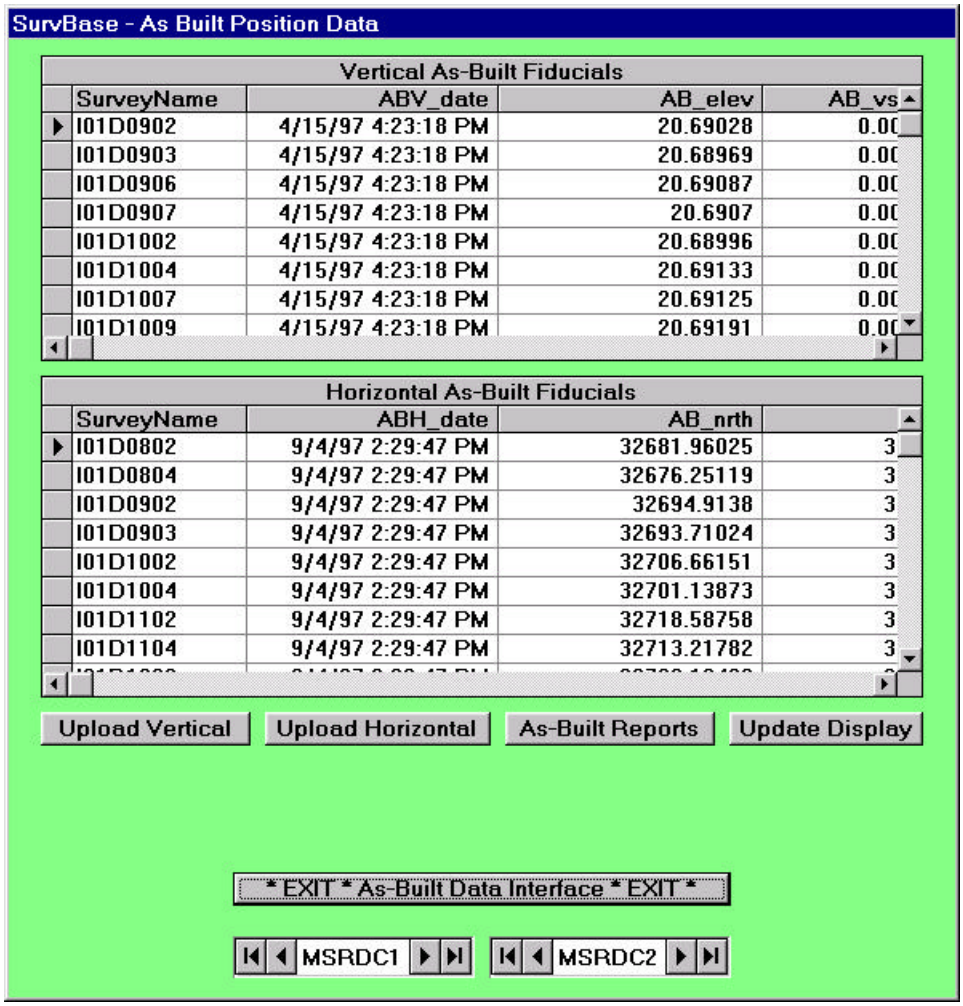

Figure 2. – RDO controls exposed at bottom of form.

Typically the command buttons on a SurvBase form contain program code that will provide a certain data handling or reporting function to the user. Each of the buttons in Figure 2, relating to as-built data, contain code that will interact with the database server.

### 7.2 RDO Code Examples

The code snippets below reflect a selection of some of the basic tools available with RDO and the amount of coding required to get full function interaction with the server.

1. Define a new RDO connection named ‘cnSurvbase’, set the database to be used on Data Source Named ‘Sybase 11.1’ as ‘survbase’ and establish the connection to the database server requiring a complete login by the user.

```
Dim cnSurvbase As New rdoConnection

With cnSurvbase
        .Connect = "Database=survbase;DSN=Sybase11.1;"
        .CursorDriver = rdUseOdbc
        .EstablishConnection rdDriverCompleteRequired, False
End With
```

2. Set the string variable SQLtxt equal to a valid SQL code statement that clears the servers transaction log for the survbase database and execute the SQL code directly on the server.

```
SQLtxt = "DUMP TRAN survbase WITH truncate_only"
cnSurvbase.Execute SQLtxt, rdExecDirect
```

3. Define a rdoResultset named rsABHorz which will contain results returned by a new rdoQuery named Query1 whose SQL statement is given by the string variable SQLtxt that has two query parameters (variables) shown as ? marks. Open a result set initially with null query parameters which will set up the query as a stored procedure on the server.

```
Dim rsABHorz As rdoResultset
Dim Query1 As New rdoQuery

SQLtxt = "SELECT * FROM AsBuilt_Horz  WHERE SurveyName LIKE ? AND
            ABH_date=?"

With Query1
        .SQL = SQLtxt
        .RowsetSize = 20000
        Set .ActiveConnection = cnSurvbase
        Set rsABHorz = .OpenResultset(rdOpenKeyset, rdConcurRowver)
End With
```

4. In the above snippet the variable SQLtxt could contain a stored procedure call.

```
SQLtxt = "get_asbuilt_3d”
```

The stored procedure ‘get_asbuilt_3d’ is defined prior to program execution in the database using the following SQL code:

```
CREATE PROCEDURE get_asbuilt_3d
SELECT AsBuilt_Horz.SurveyName, AB_nrth, AB_nsdev, AB_east, AB_esdev,
AB_elev, AB_vsdev, ABH_date, ABV_date
FROM AsBuilt_Horz, AsBuilt_Vert
WHERE AsBuilt_Horz.SurveyName = AsBuilt_Vert.SurveyName
GROUP BY AsBuilt_Horz.SurveyName, AsBuilt_Vert.SurveyName
HAVING (ABH_date=MAX(ABH_date) AND ABV_date = MAX(ABV_date))
ORDER BY AsBuilt_Horz.SurveyName
COMPUTE AVG(AB_nsdev), AVG(AB_esdev), AVG(AB_vsdev)
```

5. Set the query parameters equal to two known and previously defined string variables and re-populate the result set using the .Requery method.. Set the integer variable RecCount% equal to the count of the number of rows returned in the result set rsABHorz using the RowCount method.

```
Query1.rdoParameters(0) = SurveyName$
Query1.rdoParameters(1) = ABH_date$
rsABHorz.Requery
RecCount% = rsABHorz.RowCount
```

6. Start a controlled transaction on the server and add a new record to the current result set table (defined by 'SELECT * FROM AsBuilt_Horz' in Query1) by setting each field in the new record to a previously defined string variable and then updating the result set which will update the underlying database table. Commit the transaction to the database.

```
cnSurvbase.BeginTrans

With rsABHorz
        .AddNew
        !SurveyName = SurveyName$
        !ABH_date = ABH_date$
        !AB_nrth = AB_nrth$
        !AB_east = AB_east$
        !AB_aphi = AB_aphi$
        !AB_amaj = AB_amaj$
        !AB_bmin = AB_bmin$
        !AB_nsdev = AB_nsdev$
        !AB_esdev = AB_esdev$
        .Update
End With

cnSurvbase.CommitTrans
```

These code snippets show the power of the RDO functions in Visual Basic and the fairly straight forward coding necessary to implement their use. Once grasped, the combination of RDO and the VB programming environment enables full function client/server development with a reasonable blend of control of database interaction and ease of use for the developer. RDO is appropriately characterized as a thin layer over the ODBC API since it masks the very detailed coding required to use the API directly.

## 6.0 SURVBASE & ACCELERATOR PHYSICS

In addition to its function as an alignment data management tool for that data generated specifically by the Survey & Alignment Group including colloid or antenna measured magnet center data, the SurvBase database acts also a repository for alignment related calculation results using this data, produced by the RHIC Accelerator Physics Group (RAP). RAP results data in the SurvBase database include design values in the 'Lattice' table, magnet center statistics in the 'CenterStat' table and magnet coordinate transformation data relating to Ideals computation in

the ‘Transform’ table. These tables are managed by RAP using UNIX based C and C++ software written by the physicists using two-tier architecture similar to the SurvBase client. This illustrates the capacity of modern Relational Databases to serve multiple heterogeneous clients.

## 7.0 POTENTIAL SURVBASE FEATURES

The critical and highly specific nature of the application of data logic as related to a specific business, reveals the reasoning behind an investment in the effort necessary to develop a three-tier architecture client/server program. In the specific case of alignment data management, a great majority of the referential and internal integrity rules are the same no matter where or what type of alignment operation is underway. Naming and date/time issues are at the core of integrity requirements. A full blown implementation of a client/server application for accelerator alignment would probably be developed with a “multi-tier” architecture that would include the user interface (Tier1), a data logic server (Tier2), a computation server (Tier3) and a database server (Tier4).

The Tier3 services envisioned here would include augmented seamless links to computation and analysis software such as Star*Net, Star*Lev, AutoCAD and MS Excel. It would also include data collection device management for upload and download processes across a site computer network.

These ideas are driving the future development of SurvBase.

## APPENDIX A - SurvBase SOFTWARE & HARDWARE COMPONENTS

The following lists current software components and their version numbers:

Client Side

1. Operating System: Windows NT Version 4.0 (Build 1381: Service Pack 3)
2. Programming Language: Visual Basic 5.0 Enterprise Edition (Service Pack 2)
3. Database Interface Driver: Sybase Open Client/C Developers Kit 11.1.0
4. ODBC Driver: Sybase ODBC Driver Kit 11.1 for Windows NT
5. Hardware: Micron Millenia Pro2; 200mhz Pentium Pro; 64mb; 10mbit Ethernet

Server Side

1. Operating System : Solaris 5.5
2. Database : Sybase SQL Server 11.0.2
3. ODBC Extensions: sbt_datatype_info; sbt_datatype_info_ext; sbt_server_info
4. Hardware : Sun Sparc Server 1000; 2 cpu; 128mb; 10mbit Ethernet

Bibliography


[1] Elliott, R. & Powers, N., (Intellex)**, "**One -Tier, Two-Tier, Three-Tier, A Server: Using Technology to Solve Business Problems", http://www.pacific-electric.com/PacificElec/Product/whtpap04.htm

[2] Hemmer, F.M. (1993) "RHIC Electronic Data Collection and survey & Alignment Database" Proceedings of the Third International Workshop On Acclerator Alignment, Annecy, pp 197-

[3] Groff, J.R. and Weinberg, P.N., (Osborne McGraw Hill, 1990), Using SQL, pp. 277

[4] Paulsell, K. and Deering, B., (Sybase, Inc., 1992), Commands Reference Manual for Sybase SQL Server, pp. 2-76