Whitemarsh
Information Systems Corporation

SQL:1999 and its Impact on Data and Database Administration

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: mmgorman@wiscorp.com
Web: www.wiscorp.com

Table of Contents

Whitemarsh Information Systems Corporation 1

What's a DBMS Data Model 2

SQL 1999 3
- SQL 1999 Data Model 4
  - SQL 1999 Data Structures 5
    - SQL 1999 Data Types (really data structures) 6
      - SQL 1999 Data Type: Ref Types 7
      - SQL 1999 Data Type: Arrays 9
      - SQL 1999 Data Type: ROW Data Structures 11
      - SQL 1999 Data Type: User Defined Types 13
- SQL 1999 Foundation Components 20
  - SQL 1999 Triggers 21
  - SQL 1999 Savepoints 23
  - SQL 1999 Roles Security Enhancement 25
  - SQL 1999 Recursion 27
  - SQL 1999 Information Schema 29
- SQL 1999 Call Level Interface 31
- SQL 1999 Multi-Media Components 33
- SQL 1999 Programming Language 35
- SQL 1999/MED 37

SQL 1999 Implementations 39

Summary and Conclusions 41

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

Whitemarsh Information Systems Corporation

Founded in 1981

Owner: Michael M. Gorman

- 35+ years in data processing
- 30+ years in database
- 22+ years as Secretary of ANSI NCITS H2 Technical Committee on Database

Books, Courses, Papers, and Scenarios available from Whitemarsh website, (www.wiscorp.com)

- Data Modeling
- Database
- Database Management Systems (DBMS)
- Database Project Methodology
- DBMS Selection and Evaluation
- Information Systems Planning
- Knowledge Worker Framework
- Project Management
- Repository Selection and Evaluation
- Work Metrics

Software

- Project Management
- Metabase (metadata database repository)

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

What's a DBMS Data Model

Record Organization

- Type types of data (e.g., character, integer, decimal)
- Data Structures for each named component

Relationships

- Static: "members" known to "owners" at load or update type
- Dynamic: "members" and "owners" know of each other only at retrieval time
- Referential integrity and referential actions: a DBMS facility for both static and dynamic relationships

Operations

- Record actions (add, delete, or modify)
- Relationship actions on Owner modify and delete, and Member insert and modify

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999

- Data Model (data structure, relationships, and operations)

- SQL 1999 Parts (of interest)
  - Foundation
  - CLI
  - PSM
  - Transaction Management

- SQL:MM
  - Spatial
  - Full Text
  - Still Image

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Model

- Data Structures
- Relationships
- Operations

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Structures

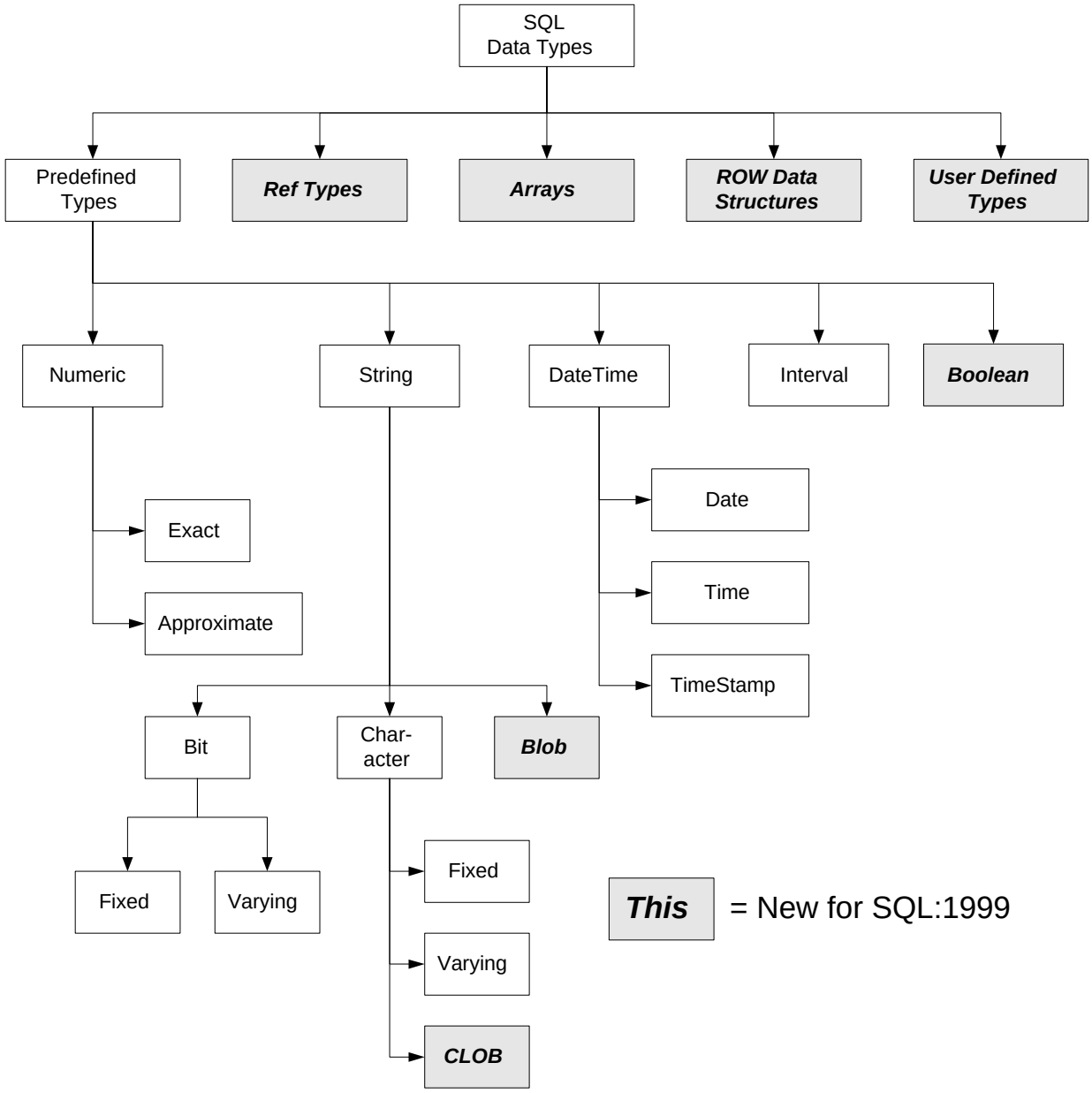
Column Data Type	Definition	SQL 1999
Single Value	Each component represents a single value such as Birthdate with the value 11/11/1987	✔
Multi-value	Each component represents multiple values such as Nicknames with values "Buddy, Guy, Mac"	✔-Note 1
Groups	Each component has subcomponents to represent single-set of values such as Address with Street-1, Street-2, City, State, Zip	✔-Note 2
Repeating Groups	Each component has subcomponents to represent multi-sets of values such as Dependents that contains subcomponents, Dependent Name, Dependent Birth date, Dependent SSN.	✔-Note 3
Nested Repeating Groups	Employee (Dependents (Hobbies))	✔-Note 4

Notes:

1	Arrays as a data type for a column
2	ROW data structure of a column
3	ROW data structure for a column wherein each Row structure field has the data type, ARRAY
4	ROW data structure for a column with contained ARRAYs where each ARRAY element is a ROW data structure with contained ARRAYs where each ARRAY element , etc....

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Types (really data structures)



Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: Ref Types

References to instances of structured types that can be used wherever other types are used

Representation

- User generated (REF USING <predefined type>)
- System generated (REF IS SYSTEM GENERATED)
- Derived from a list of attributes (REF (<attribute definition>s)

```
<attribute definition> ::= <attribute name>
        <data type>
        [ <reference scope check> ]
        [ <attribute default> ]
        [ <collate clause> ]
<attribute default> ::= <default clause>
```

Example

```
CREATE SCHEMA s1

CREATE TYPE employee AS
    (lastname       CHARACTER VARYING (30),
    firstname       CHARACTER VARYING (30),
    manager         REF (employee) SCOPE EVERY EXISTING TABLE ) ;

CREATE TABLE employees OF employee
( empref REF (employee) VALUES ARE SYSTEM GENERATED ) ;

CREATE TABLE project_teams
    ( team_name     CHARACTER VARYING (30),
    team_leader     REF (employee) SCOPE (employees),
    team            REF (employee) SCOPE (employees) ARRAY [10],
    reports         ROW ( frequency CHARACTER VARYING (30),
    report_to       REF (employee) SCOPE (employees)
```

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL:1999 Component: Ref Types	
Impact Area	**Impact**
Database design	Creates the ability to have “pointer” base relationships to objects from different classes.... Employee REF to Person
Data administration	Increased analysis of collections and groups of data
Database administration	Performance is affected by references to objects on devices of unknown and possibly inconsistent performance.
Database application	Should simplify the application design’s need to know foreign key values

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: Arrays

```
CREATE TABLE reports
        (id INTEGER,
        authors VARCHAR(15) ARRAY[20],
        title VARCHAR(100),
        abstract FullText)
```

Appropriate DML operations

```
INSERT INTO reports (id, authors, title)

VALUES (10, ARRAY ['Date', 'Darwen'], 'A Guide to the SQL Standard')
Standard')
```

Access to array elements

- By ordinal position
- Declarative (i.e. query) facility
- Implicitly transforms array into table
- Selection by element content and/or position
- Unnesting

Examples:

```
SELECT id, authors[1] AS name FROM reports

SELECT r.id, a.name FROM reports AS r, UNNEST (r.authors) AS a (name)
```

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL:1999 Component: Arrays	
Impact Area	**Impact**
Database design	Creates the ability to have ordered multi-valued items within a column such as nick-names or phone numbers. Can be coupled with Ref Types, so that makes life very "interesting." Greater separation between "logical" and physical design. Eliminates the need for either blatant denormalization or single column tables.
Data administration	Decreased design efforts for a particular database. But, if order or position is "information bearing" then that information needs to be defined and stored somewhere.
Database administration	Should increase performance. Less relationship processing. Less performance tuning. Will have negative data exchange or replication impact with other DBMSs not supporting arrays.
Database application	Application will have to have extra navigation language/facilities build in to process the elements of the array.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: ROW Data Structures

```
CREATE TABLE employees
        (name CHAR (40),
                address ROW ( street CHAR (30),
                                city CHAR (20),
                                zip ROW( original CHAR (5),
                                        plus4 CHAR (4)));


INSERT INTO employees
        VALUES (‘John Doe’, (‘2225 Coral Drive’, ‘San Jose’, (‘95124’, ‘2347’)));
```

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL:1999 Component: ROW data types	
Impact Area	**Impact**
Database design	Creates the ability to have a defined collection or data structure of fields within the ROW type of a particular column of a table. Examples include Address. There are no restrictions on the quantity of levels or on the data types that can be attached to the "fields" within the ROW data type. Enables collapsing of collections of database tables into a well defined collection of contained structures.
Data administration	More difficult administration if different tables have the same defined contained ROW type. Forces the real separation and then continuous mapping between the Data Administrator's data model (specified data model) and the Database administrator's data model (implemented data model).
Database administration	May well cause the DBA's data model to be quite different from the DA's data model. Makes the ability to have data replication and data sharing more difficult as the same concept may be designed into different tables very differently. Different DBMSs are likely to implement this capability very differently. Some by creating real nested data structures (fastest) to merely unwrapping the nested structures into hidden but separate physical tables (slowest).
Database application	Increased programming logic within the application program to navigate the contained data structure. If the "same" data is designed differently in different databases then the accessing program's logic will have to be quite different. Elevates the need for and value of an SQL view.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: User Defined Types

- User-defined data types

  User-defined, named types representing entities.

  For example, employee, project, money, polygon, image, text, language, format, ...

- User-defined methods and functions (operators)

  User-defined operation representing the behavior of entities in the application domain.

  For example, hire, appraisal, convert, area, length, contains, ranking, ...

- Definition:

User-defined data type
- Name
- Representation
- Relationship to other types

User-defined method (and function)
- Name
- Signature (i.e., parameter list)
- Result
- Implementation

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: UDT–Distinct Types

```
CREATE TABLE RoomProperties
        ( RoomID CHAR(10),
        RoomLength          INTEGER,
        RoomWidth           INTEGER,
        RoomArea            INTEGER,
        RoomPerimeter       INTEGER );


UPDATE RoomProperties
SET RoomArea = RoomLength;
```

No error results because the column names are all traditional data types that contain no enforced semantics other than fundamental nature of the value....

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

But with Distinct Types,

First Create the Distinct Types for the Columns

```
CREATE TYPE plan.roomtype AS CHAR(10) FINAL;

CREATE TYPE plan.meters AS INTEGER FINAL;

CREATE TYPE plan.squaremeters AS INTEGER FINAL;
```

Then Create the Table

```
CREATE TABLE RoomTable (
        RoomID plan.roomtype,          plan.roomtype
        RoomLength                     plan.meters,
        RoomWidth                      plan.meters,
        RoomPerimeter                  plan.meters,
        RoomArea                       plan.squaremeters);


UPDATE RoomTable
        SET RoomArea = RoomLength;
```

You get an ERROR because the data type of RoomArea <> data type of Room

```
UPDATE RoomTable
SET RoomLength =RoomWidth;
```

NO ERROR RESULTS because the data types are the same, plan.meters,

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

User Defined Methods

- **What are methods?**--SQL-invoked functions "attached" to user-defined types

- How are they different from functions?

  - Implicit SELF parameter (called subject parameter)
  - Two-step creation process: signature and body specified separately.
  - Must be created in the type's schema
  - Different style of invocation (UDT value.method(...))

Example:

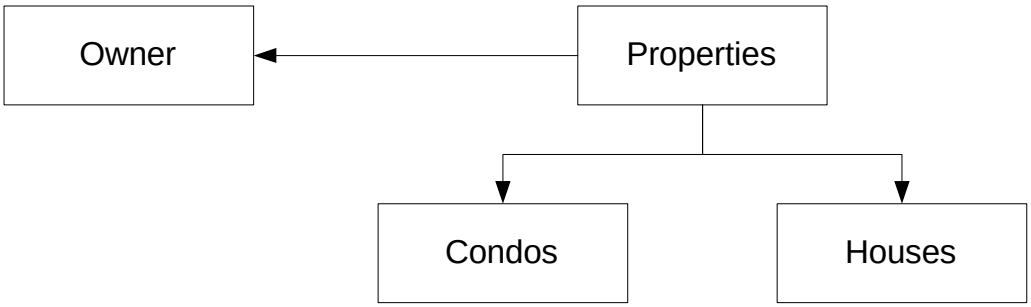
```
CREATE TYPE employee AS
        (name CHAR(40),
        base_salary DECIMAL(9,2),
        bonus DECIMAL(9,2))
        INSTANTIABLE NOT FINAL
        METHOD salary() RETURNS DECIMAL(9,2);

CREATE METHOD salary() FOR employee
        BEGIN
        ....
        END;
```

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Data Type: UDT Structured Types, Sub Tables

CREATE TYPE person ... NOT FINAL
CREATE TYPE real_estate ... NOT FINAL
CREATE TYPE condo UNDER real_estate ... NOT FINAL
CREATE TYPE house UNDER real_estate ... NOT FINAL

CREATE TABLE people OF person (...)
CREATE TABLE properties OF real_estate
CREATE TABLE condos OF condo **UNDER** properties
CREATE TABLE houses OF house **UNDER** properties



NB: Subtables are MERELY a method of partitioning a BIG table into subtyped subordinate tables. For each SuperTable row there is AT MOST one row from each subtable, but,

Count(Properties) >= Sum (Count(Condos) + Count(Houses))

Example: If there were 100 properties then if there were 75 houses there could only be a maximum of 25 condos...

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: User Defined Types	
Impact Area	**Impact**
Database design	Greatly expands the scope of database definition from just simple columns within tables to complex nested data structures that have both strong data types and contained methods.
Data administration	Requires significantly increased knowledge of full scope of the database and its general application use to correctly include complex data types where appropriate with the database's design.
Database administration	Expands job to included knowledge of complex nested data structures and the ability to understand, debug, test, and maintain embedded processes within the database design.
Database application	Should simplify the database application as common logic and processing is factored out and installed within the DBMS firewall.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Relationship Types

Name	Example	SQL 1999
One-to-many	Employee to dependents	✔-Note 1
Owner-multiple-member	Territory contains salesmen and customers	✔-Note 2
Singular-one-member	Top performing employees	✔-Note 1
Singular-multiple-member	Top performing current, former, part-time, and retired employees	✔-Note 3
Recursive	Organization contains organization	✔-Note 4
Many-to-many	Automobiles and owners	✔-Note 5
One-to-one	Table and its primary key	✔-Note 6
Inferential	Many houses each with a location, and then buyer with desired location	✔-Note 7

Notes:

1	Traditional relational data model (SQL:1986, 1989, & 1992)
2	Implemented as a ROW(TerritoryId, Salesman REF (SalesmanId), Cust_Id (integer) ARRAY[500]),
3	Developed using Subtables where Employees are partitioned off into their common columns (employee) and their unique columns (current, former, part-time, and retired)
4	Recursion operations built into the language (WITH RECURSION...)
5	Cross joins from within elements of ARRAYS contained as data types of column in different tables
6	Effectively as tables and subtables. Most directly with UNIQUE Fkey
7	A single valued non-primary key Location within House and same for Buyer

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Foundation Components

- Tables that have been enhanced to support new built-in data types (boolean, extensions to character sets, translations, and collations)
- BLOB and CLOB data types
- User Defined Types (behavior, an encapsulated internal structure, and access characteristics of public, protected, or private)
- Array
- ROW (table person (SSN, name ROW(first, middle, last), address ROW(street, city, state, zip ROW(four, five)))))
- User Defined Functions
- Predicate extensions (for all, for some, similar to, cursor extensions, null values, assertions, view updatability, joins)
- Triggers
  - Different triggering events, update, delete, and insert
  - Optional condition
  - Activation time: before and after
  - Multiple statement action
  - Several triggers per table
  - User-defined ordering
  - Condition and multiple statement action per each row or per statement
- Roles (enhancements to security), & Save-points
- Recursion
- Information Schema

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Triggers

- Triggers provide automatic execution of a set of SQL statements when a specific data change operation (UPDATE, INSERT, DELETE) occurs

- Bring application logic into the database

- Transforms a passive to active DBMS

  - Benefits of triggers include:
  - Code reuse
  - Faster application development
  - Easier Maintenance
  - Guaranteed enforcement of business rules

- Common Use of Triggers

  - Enforce "transitional" business rules
  - Validate input data
  - Generate new values for inserted / updated rows
  - Cross-reference other tables
  - Maintain audit, summary or mirror data in other tables
  - Support "alerts"
  - E-mail notification
  - Initiate external actions

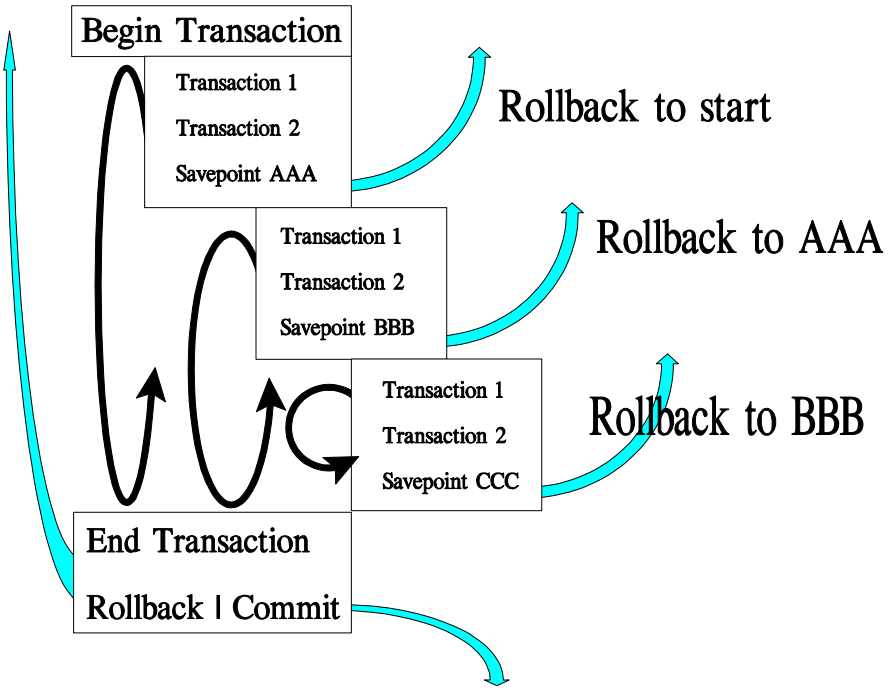
Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: Triggers	
Impact Area	**Impact**
Database design	Creates the ability to have stored procedures that are activated on the occurrence of a database state change. Expands the effort of database design from just data to contained process that transcends application programs. Triggered actions can range from simple declarative actions to many pages of SQL programming language statements.
Data administration	Data administration must now assert its role in defining, programming, debugging, and maintaining the business rules that are to be inside the DBMS firewall. These embedded programs must be specified in ANSI SQL so that it can be mapped to the different DBMS implementations.
Database administration	Database administration must now be intimately involved in trigger process specification, implementation, debugging, and maintenance. Trigger implementations of different DBMSs are likely to be different and will require mapping from the Data Administrator trigger process specifications. DBA will get involved in test data generation, pre- and post-state specification, etc.
Database application	Should greatly simplify database applications as critical business rules are moved inside the DBMS firewall.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Savepoints

- Behaves like single-nested subtransactions
- ROLLBACK TO SAVEPOINT allows “partial rollback” of transaction
- RELEASE SAVEPOINT acts like tentative commit of part of transaction



Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: Savepoints	
Impact Area	**Impact**
Database design	None really
Data administration	None really
Database administration	Creates the ability to have interim process stream "almost-commits" to the database via the DBMS. Enables batches of updates to occur and then accomplish some sophisticated processing to then determine to roll-back or to continue.
Database application	Moves some of the safety logic from the computer program and from its creation of program-based save points to the control of the DBMS

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Roles Security Enhancement

New in SQL99; benefits:

- Simplifies definition of complex sets of privileges
- Roles are created

Note: definition of users implementation-defined

Example:

- CREATE ROLE Auditor

- CREATE ROLE AuditorGeneral

Roles may be assigned to users & roles

- GRANT Auditor TO AuditorGeneral WITH ADMIN OPTION
- GRANTED BY CURRENT ROLE
- GRANT Auditor TO Smith

Controllable whether to grant as user or role

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: Roles based Security	
Impact Area	**Impact**
Database design	None really
Data administration	None really
Database administration	Enables sophisticated security of groups of persons performing similar transactions. Supports the need for a security administrator within the DBMS environment.
Database application	Moves DBMS based aspects of security from the application program, 4GL, or query to the DBMS for central definition and administration.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Recursion

What is recursive SQL?

- self-referencing table expressions
- self-referencing views

Why use recursion?

- Bill of material processing
- Network traversals (e.g. airline routing)

Functionality and performance benefits

- Challenge: integration into SQL
- Advanced recursion (e.g. mutual recursion)
- Integration with different forms of joins
- Allows for duplicates
- Graph traversal in "depth first" or "breadth first"
- Cycle control

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: Recursion	
Impact Area	**Impact**
Database design	Creates the ability to have SQL language directly process hierarchically organized data from one table. This is a critical capability for must commercial and MIS type applications. Prior to this command-set, the user had to create the exact processing logic to accomplish both depth first and/or width first processing.
Data administration	Saves extra database design effort to match the needs of different types of recursions.
Database administration	Saves time on testing and debugging. Additionally, enables the DBMS to perform recursion on the server-side rather in the individual client application.
Database application	Makes it simpler to create and code as this major requirement is pushed to the

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Information Schema

A set of views describing the metadata contained in a catalog

- Exist in the INFORMATION_SCHEMA schema
- Are fully defined (column names, data types, and semantics)
- May be queried by users
- Are read-only
- Reflect database objects that the user owns or for which the user has some privilege
  - TABLES
  - COLUMNS
  - VIEWS
  - DOMAINS
  - etc

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: Information Schema	
Impact Area	**Impact**
Database design	None really.
Data administration	Creates the ability to have three levels of abstraction in every database: fundamental, schema objects, and schema object instances
Database administration	None really
Database application	Database is able to be queried as to its exact schema contents to support formulation of dynamic SQL commands and to ease development of applications that need federated and distributed databases

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Call Level Interface

The CLI specification contains more than 50 different call specifications that address:

- Connection control to SQL servers
- Allocate and de-allocate resource
- Execute SQL statements
- Obtain diagnostic information
- Control transaction termination
- Obtain information about implementation

It also contains Resource Management Handle routines for:

- Environment
- Connection
- Statement
- Context

The CLI also contains a Descriptor Area that accommodates:

- Application parameter
- Application row
- Implementation parameter
- Implementation row

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL1999 Component: Call Level Interface (SQL subset of ODBC)	
Impact Area	**Impact**
Database design	None really
Data administration	None really
Database administration	Allows for the integration of data from multiple DBMSs through SQL/CLI
Database application	Allows the inclusion of 3rd party software without the knowledge of exactly which DBMS the third party software was created for.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Multi-Media Components

- Framework (Part 1)
- Full Text (Part 2)
- Spatial (Part 3)
- Still Image (Part 5)

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL:1999 Component: Multi-media components (Text, Spatial, Image)	
Impact Area	**Impact**
Database design	Creates the ability to integrate multiple classes of data that are traditionally non-database such as full text, geometric, and large images or sound.
Data administration	Expands the domain of the data designs and data standardization efforts of Data Administration.
Database administration	Expands the domain of the data controlled by the DBA
Database application	Enables applications to integrate data from multiple types and classes. E.g., the catalogues of books (traditionally database) with the actual text of the books.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Programming Language

Procedural Extensions

- Improve performance in centralized and client/server environments
- Multiple SQL statements in a single EXEC SQL
- Multi-statement procedures, functions, and methods

Gives great power to DBMS

- Several, new control statements (procedural language extension)
- (begin/end block, assignment, call, case, if, loop, for, signal/resignal,
- variables, exception handling)

SQL-only implementation of complex functions

- Without worrying about security ("firewall")
- Without worrying about performance ("local call")

SQL-only implementation of class libraries

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: SQL Programming Language	
Impact Area	**Impact**
Database design	Greatly expands the scope and efforts of database design. Once restricted to just data, database design now includes complete specification of the processes that are common to multiple applications. The programming language permits server-side, application independent processing of SQL data.
Data administration	Data administration must now assert its role in defining, programming, debugging, and maintaining the business rules that are to be inside the DBMS firewall. These embedded programs must be specified in ANSI SQL so that it can be mapped to the different DBMS implementations.
Database administration	Database administration must now be intimately involved in process specification, implementation, debugging, and maintenance. Implementations of the SQL programming language from different DBMSs are likely to be different and will require mapping from the Data Administrator's trigger process specifications. DBA will get involved in test data generation, pre- and post-state specification, etc.
Database application	Should greatly simplify database applications as critical business rules are moved inside the DBMS firewall.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999/MED

- Management of External Data
- Seen as way to give SQL access to non-SQL data (e.g., flat files, even sensors)
- Foreign tables, abstract LOBs: SQL API
- Federated database?
- (Non-final) Committee Draft late 1999

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Component: MED	
Impact Area	**Impact**
Database design	Creates the ability to integrate classes of data that are traditionally outside the domain of traditional database. Included would be COBOL based file structures, ASCII file structures, and the like. These files' structures will have to be determined and made known to the DBMS.
Data administration	Expands the domain of the data designs and data standardization efforts of Data Administration.
Database administration	Expands the domain of the data controlled by the DBA
Database application	Enables applications to integrate data that has been created from non-DBMS methods to be incorporated into DBMS-based applications.

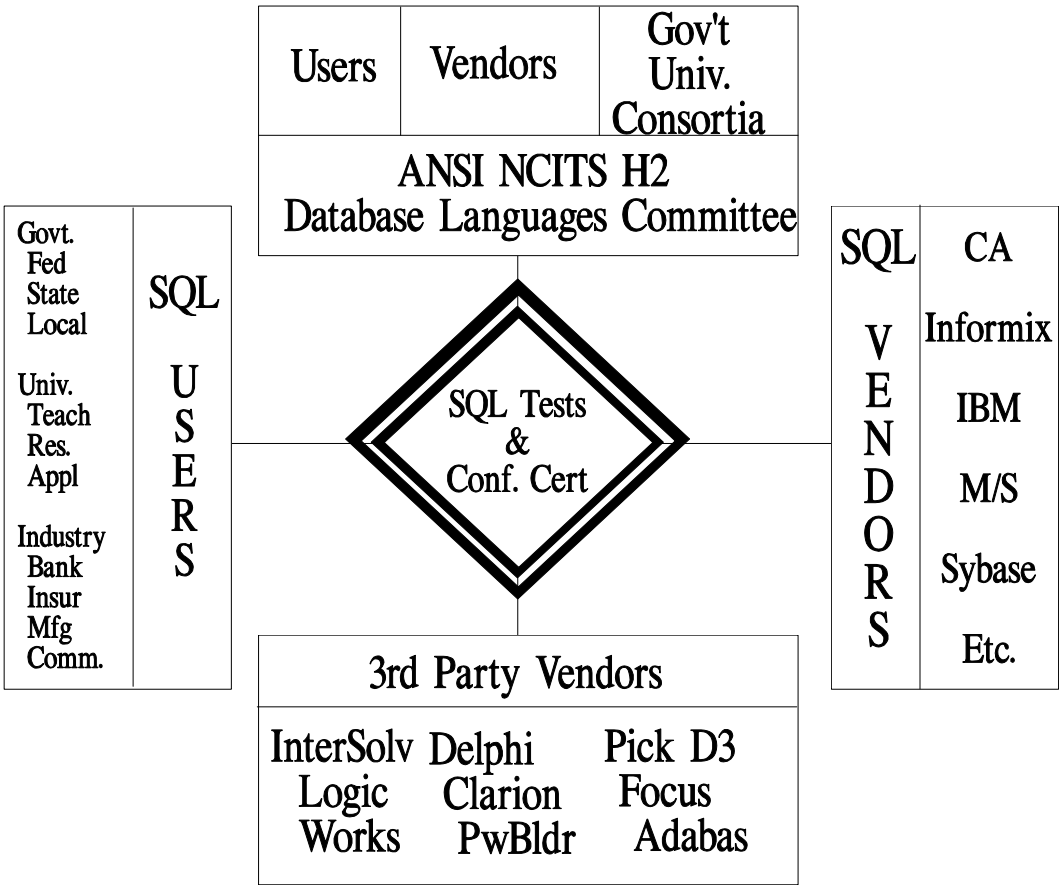
Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

SQL 1999 Implementations

- Most vendors say “about 2 or 3 product cycles to conform to Core SQL 1999”
- Suggests late 2001 for conforming products
- A few vendors claim they will conform sooner
- Vendors will choose packages based on their perception of marketplace needs
- Vendors will implement selected other features as needed

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

And, How Will *You* Know Who Conforms???



You won't because there are no conformance tests or suites, and nobody is undertaking to accomplish them.....

NIST created and performed conformance testing for the U.S. Federal Government from 1989 through 1996. NIST stopped to "save" $600K per year. Savings to Fed was over $35,000,000 per year.

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

Summary and Conclusions

Simply stated, the SQL 1999 language defines a unique data model. It contains:

- The ability to model CODASYL sets,
- Many of the natural data clustering features of the hierarchical data model,
- Explicit many-to-many and inferential relationships like the independent logical file data model, and finally,
- The unique ability to directly model recursive relationships.

And, SQL 1999 contains:

- A full programming language (without presentation layer) for complete embedded process capabilities

Support for:

- Spatial
- Full Text
- Still Image

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

Feature that through standard or non-standard ways accomplishes:	Percent Distribution Across SQL Vendors			
	1. Have Already Imple-mented	**2. Currently Imple-menting**	**3. Investi-gating**	**4. Definitely Not Imple-menting**
Nested data structures within columns (e.g., array, Ref Types, ROW, UDTs, et al)	100			
BLOB, CLOB	100			
Methods within Columns	90	5	5	
Triggers	100			
Savepoints	90	5	5	
Recursion	35		55	10
SQL/CLI	90		10	
SQL/PSM	90	10		
SQL/Full Text	80			20
SQL/Spatial	55		15	30
SQL/MED	55	15	15	15
Export data through XML	40	15	30	15
Export schema et al through XML	40	10	40	10

SQL Vendors on the ANSI NCITS H2 Technical Committee on Database:

Compaq, Computer Associates, Informix, Corel, International Business Machines, Intersystems, Microsoft, NCR, Oracle, Pervasive, Progress, and Sybase

Copyright 2000, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved