# Data Model and Relational Database Design for the New England Water-Use Data System (NEWUDS)

*By* Steven Tessler

## Abstract

The New England Water-Use Data System (NEWUDS) is a database for the storage and retrieval of water-use data. NEWUDS can handle data covering many facets of water use, including (1) tracking various types of water-use activities (withdrawals, returns, transfers, distributions, consumptive-use, wastewater collection, and treatment); (2) the description, classification and location of places and organizations involved in water-use activities; (3) details about measured or estimated volumes of water associated with water-use activities; and (4) information about data sources and water resources associated with water use. In NEWUDS, each water transaction occurs unidirectionally between two site objects, and the sites and conveyances form a water network. The core entities in the NEWUDS model are site, conveyance, transaction/rate, location, and owner. Other important entities include water resources (used for withdrawals and returns), data sources, and aliases. Multiple water-exchange estimates can be stored for individual transactions based on different methods or data sources. Storage of user-defined details is accommodated for several of the main entities. Numerous tables containing classification terms facilitate detailed descriptions of data items and can be used for routine or custom data summarization. NEWUDS handles single-user and aggregate-user water-use data, can be used for large or small water-network projects, and is available as a stand-alone Microsoft® Access database structure. Users can customize and extend the database, link it to other databases, or implement the design in other relational database applications.

## INTRODUCTION

In many areas of New England, withdrawals of freshwater are approaching the operational capacities of developed water supplies. Local, State, and Federal agencies need data on all aspects of water use to develop comprehensive water-resource management plans and to make decisions regarding water-supply development and requirements for water conservation measures. Sound decisions about the development of new water supplies and the efficient use of existing supplies require current, accurate, and complete information on what happens to water from points of withdrawal to points of return flow. Decisions such as whether to expand withdrawals in one area or limit them in another need to be supported by a geographic inventory of existing withdrawals, interbasin transfers, leakage, consumptive use, and return flow. Water-use decisions also may affect the environment, often directly, when overuse of surface-water sources leads to reductions in streamflow and changes in habitat and biological communities, or when streamflow consists primarily of treated effluent of varying quality during part of a year. An effective water-resource management plan is contingent upon the data provided by a comprehensive water-use program (Horn and Craft, 1991).

Water use, in the broadest sense, pertains to the interaction between human activity and the hydrologic cycle (Solley and others, 1998). Water use begins when water is diverted or withdrawn from surface- or ground-water sources and conveyed to a place of use. A withdrawal is made by a user or by a public supply system, which conveys the water to users through a distribution system. Water use by a user or group of users refers to water that is used for a specific purpose, such as for domestic functions, irrigation, or industrial processing. Consumptive use refers to water that evaporates or is incorporated into a product during use. Water in a distribution system may be leaked back into the hydrologic system or put to public use, such as sanitation, fire fighting, or hydrant flushing. After use, wastewater is conveyed to a treatment facility and return flow, or is returned directly to the hydrologic system through septic systems. Wastewater in a collection system may be leaked back to the hydrologic system, and may receive water from surface runoff (inflow) or ground water (infiltration). For a more comprehensive description of water use, the reader is referred to the National Handbook of Recommended Methods for Water Data Acquisition—Chapter 11, Water Use (accessed October 20, 2001, on the World Wide Web at URL http://water.usgs.gov/pubs/chapter11/).

In New England, water-use data have been collected and analyzed since the 1970s and the data have been stored in a variety of ways. Beginning in about 1990, data were stored in the Site Specific Water-Use Data Systems (SWUDS) developed by the U.S. Geological Survey (USGS) cooperative water-use program to store and retrieve water-use data collected or compiled at the site-specific level. Later, data were stored in a series of spreadsheets that were easier to use than SWUDS. Most recently, the New England Districts of USGS (Connecticut, Massachusetts–Rhode Island, New Hampshire–Vermont, and Maine) coordinated development of a database that would store the existing water-use data and promote efficient analysis and retrieval of water-use data in support of project activities throughout New England. An internal USGS workgroup was formed in 1997 to develop a PC-based, stand-alone water-use data system for the New England Districts. Goals for this new database were to facilitate uniform data description and quality among the districts, provide a more flexible alternative to the optional national USGS database, be useful to employees new to water-use data, and be used for small, focused projects at the watershed scale. Workgroup members included water-use specialists from each District, an area water-use specialist, a management representative, a facilitator, and a database specialist. The workgroup discussed characteristics of water-use data, classification schemes, ancillary data, storage and retrieval needs for standard reporting, and the features of a system that would allow efficient and rapid examination of the data to provide enhanced customer service.

The New England Water-Use Data System (NEWUDS) was developed to store water-use information that allows water to be tracked from a point of a water-use activity, such as withdrawal from a resource, to a second point of water-use activity, such as delivery to a user. The links between water-use activities can start from the initial withdrawal from the resource and end with the point of final return. NEWUDS has the following features:

- It is constructed from a conveyance-based data model rather than a site-based data model, thus promoting and encouraging a water network approach to water-use data storage and investigation.
- It handles both single-user and aggregate-user water-use data in a single data model.
- It is implemented as a stand-alone (and portable) database in Microsoft® Access (MS Access) and therefore accessible to a large number of potential users. The design can be recreated in any other relational database management system with some modification.
- It can be used for large projects (states and regions) and small, focused projects (such as watershed studies).
- It is fully open to customization and extension.

Throughout the remainder of this document, logical entities are denoted by capitalization (for example, Site and Owner); tables, fields and name parts are shown in italics (for example, the table *tblSite*, the field *SiteTypeCategory*, and the suffix *_ID*); example data values for fields are shown within quotes (for example, *SiteTypeCategory* = “treatment”); and query names are shown in boldface (for example, **qryRateUnitConversionFactor**).

## Purpose and Scope

The purpose of this document is to describe the NEWUDS database by describing the NEWUDS data model and its physical implementation in MS Access. Another report (Horn, 2002) describes how to represent water-use activities in a form that can be completely and accurately represented in the database and subsequently retrieved to meet user needs. This design report is intended for readers who have some background in the design or use of relational databases. Readers who are unfamiliar with data models or relational database design concepts are referred to the numerous books available on those subjects (for example, Fleming and von Halle, 1989; Hernandez, 1997; Roman, 1999), or to the Federal data modeling standard document FIPS 184 (National Institute of Standards and Technology, 1993). This report is designed to meet the following user needs:

- to get an introduction to the NEWUDS data model;
- to evaluate the NEWUDS storage structure for their use;
- to customize a copy of NEWUDS;

- to link NEWUDS to other databases (such as geographic information systems, well- and stream-gage information, or water-quality information); and
- to build software applications servicing NEWUDS for data entry, data analysis, or reporting.

Although the order of topics leads from conceptual to detailed information, some readers may prefer to read the document in a different order, or to identify a section of immediate interest. The headings for the remaining sections in the document are listed below with a brief summary of the contents.

Development of the NEWUDS Data Model—a list of requirements that the database structure had to meet, the modeling process, and the software used.

NEWUDS Design Principles—the conventions used in diagrams illustrating tables, fields, and relationships [in this report, the term relationship refers to an association between two entities or between instances of the same entity (National Institute of Standards and Technology, 1993)]; normalization goals; key structures; rationale for extensive use of domain tables; functional table types; and naming standards.

The NEWUDS Data Model—definition and description of the core logical entities and relationships that serve as the conceptual foundation for the NEWUDS data model, description of tables and relationships constituting the core of the NEWUDS database, and individual descriptions of subject areas and accessory tables.

Operational Issues and Procedures—items related to working with data within NEWUDS, including table loading order, supplied maintenance and update queries, and using standardized views as templates for generalized or custom data extraction and exploration activities.

Customizing and Extending the Data Architecture—recommended methods for adding or linking new data fields or tables to NEWUDS without compromising the base structure.

Conclusions—a brief summary of NEWUDS and some advantages of using the database.

References—a list of references cited in the document.

Appendixes—Appendix 1 contains the NEWUDS Data Dictionary, which lists and describes each table and field and their main properties; Appendix 2 contains domain table listings that show the classification and descriptive terms used to uniformly define data records within NEWUDS.

## Acknowledgments

The NEWUDS project would not have been possible without the interest and direct support of the New England District Chiefs: James Campbell, Derrill Cowing, Virginia deLima, Robert Lent, Brian Mriazik, and Wayne Sonntag. The author acknowledges the significant contributions of Marilee Horn in the initial conceptual design phase and of the following USGS workgroup members and users toward the development and finalization of the database design, testing, and implementation of the database: Marilee Horn and Laura Medalie (New Hampshire–Vermont District), Timothy Frick (Connecticut District), and Lisa Bratton (Massachusetts–Rhode Island District). The author also thanks Barbara Korzendorfer for her thoughtful editorial recommendations that improved the final report.

# DEVELOPMENT OF THE NEWUDS DATA MODEL

Creating a model that accurately describes the structure of a set of data is a precursor to any correctly designed database (Fleming and von Halle, 1989). Data modeling is the formal process of analyzing and reducing descriptions of information into separate data components, establishing the nature and direction of relationships between those components, and thereby building a structure for the data that automatically enforces the rules needed to maintain data integrity. A logical data model recognizes change as part of data management and provides a generic structure that permits later extensions without affecting the validity of data already in the database. A data model also is used to create and maintain documentation of all elements in the database, and thus provides a common language and reference for all users. In addition, a data model is used as a template for implementation of a physical database design and provides a guarantee that data entered into the database meet a predetermined level of detail and accuracy.

## NEWUDS Requirements

A set of data and user requirements is used to clarify the purpose and scope of the database, to guide the development of the database structure, and to serve as final checkpoints for the end product. The following list states the general requirements for the water-use database design.

- Store all data that are traditionally collected and used for the USGS National Water-Use Program 5-year compilation reports.
- Allow for the storage and retrieval of information concerning single-user and aggregate-user water use.
- Store classification information at multiple levels concerning the uses and suppliers/users of water to allow for flexibility in comparisons and aggregations.
- Store a wide range of location information to allow data to be summarized in different spatial contexts.
- Allow for storage of data on water exchange (volume per unit time) in a variety of measurement and reporting units that cover a range of transaction time scales (for example, daily, monthly, or annual data). It must also facilitate conversion of the data to a common unit scale so that data retrievals are in uniform units.
- Allow for the storage of alternative estimates of individual water transactions based on different determination methods or data sources.
- Identify sources of data and allow contact information to be stored.
- Allow for alternative identification and naming schemes for the main objects listed in the database.
- Allow user-defined details to be optionally added for major objects in the database.
- Guarantee the integrity of the data after data entry is completed.

## The Modeling Process

Development of the NEWUDS database began with a series of discussions about water-use terms and the logical groupings and relationships between information elements. The major data objects were identified and ways were developed to represent them in an information system. When a general water-use concept could be realized in more than one data structure, the alternatives were presented, the implications of each were discussed, and the contextually accurate structure determined and incorporated into the model. Exceptions to data generalizations and required/optional data elements were identified, and the emerging model was evaluated and revised. This process continued in an iterative fashion until all the information elements implied in the NEWUDS requirements were set in the data structure, and the factual statements represented by the relationships between all data objects were approved as accurate by the workgroup.

Design diagrams and listings of table/field properties (data dictionaries) were often used during modeling discussions or reviewed between meetings. Because water-use specialists participated in the NEWUDS design phase, they ensured that it contained those specific, categorical, and classification parameters needed to store, describe, validate, list, summarize, and aggregate their water-use information. In addition, a guiding principle was that the data needed to provide answers to what, where, how, and when questions used in water-use analysis should be in a form that is familiar to the user, in terms that are well defined and commonly used.

## Modeling Software

The NEWUDS database was designed using data-modeling tools. Initially, xCASE for Jet (MS Access) version 2.5 (RESolution, Ltd.) was used, but ERwin version 3.5.2 (Platinum Technologies) was used for the bulk of the work. Data-modeling tools facilitate rapid design and modification and invisibly handle many details of the data structures that can affect the quality and integrity of data. ERwin can model and generate physical databases for a number of relational database-management systems. For MS Access, ERwin provided editing capabilities for application-specific data properties through a physical model based on the logical design. After the NEWUDS data model was completed, ERwin was used to create the physical implementation of the database in MS Access by providing table and field identities, discrete properties (field type, size, default values), relationships, and all key field constraints that ensure data integrity.

## NEWUDS DESIGN PRINCIPLES

Building a database and communicating its features are enhanced by establishing certain rules of construction and standards for uniformity. Design principles provide standardized rules for creating and assembling the database and facilitate communication about how different elements of the database function together. This section describes the method used for the visualization of database objects, goals for table construction and establishing formal relationships, the rationale for extensive use of domain tables, and naming standards applied to tables and fields.

### Entity/Relationship Diagramming Conventions

Entity/Relationship (E/R) diagrams are used to visualize database designs, and these diagrams are presented throughout this report to describe the NEWUDS structure. Several different display and notation methods are commonly used for E/R diagrams, and all allow graphical/text representation of entities, relationships, and attributes. The diagrams in this document use the Integration Definition for Information Modeling (IDEF1X) standard as defined in National Institute of Standards and Technology (1993) and as implemented in the ERwin modeling software. An E/R diagram showing the complete set of relationships between Owners and Addresses illustrates several diagramming conventions (fig. 1).

A box is used to denote an entity, which equates to a single table in the physical database implementation. Each entity box has its corresponding table name at the top. Within the box are one or more entity attributes, which equate to fields in the physical database. Lines connecting entities represent their defined relationships. Lines signify that the entities share a key field between them and can be joined in the physical database through the relationship.

Each attribute within a table is shown with its assigned data type for the MS Access implementation of NEWUDS. The primary key is the field or group of fields that uniquely identify a record in the table. The primary key (PK) for each entity (composed of one or more attributes) is listed at the top of the attribute list within the entity box, and is separated from the non-PK attributes by a horizontal line. When a primary key from one table (the parent) is used in another table (the child) to create a defined relationship, the field is called a foreign key in the child table and has an “(FK)” designation in the diagram.

If a foreign key field also is part of the primary key in the child table, the relationship is said to be strong and the relationship line is solid in the diagram. If the foreign key is a non-PK (descriptive) attribute of the child table, the relationship is said to be weak and

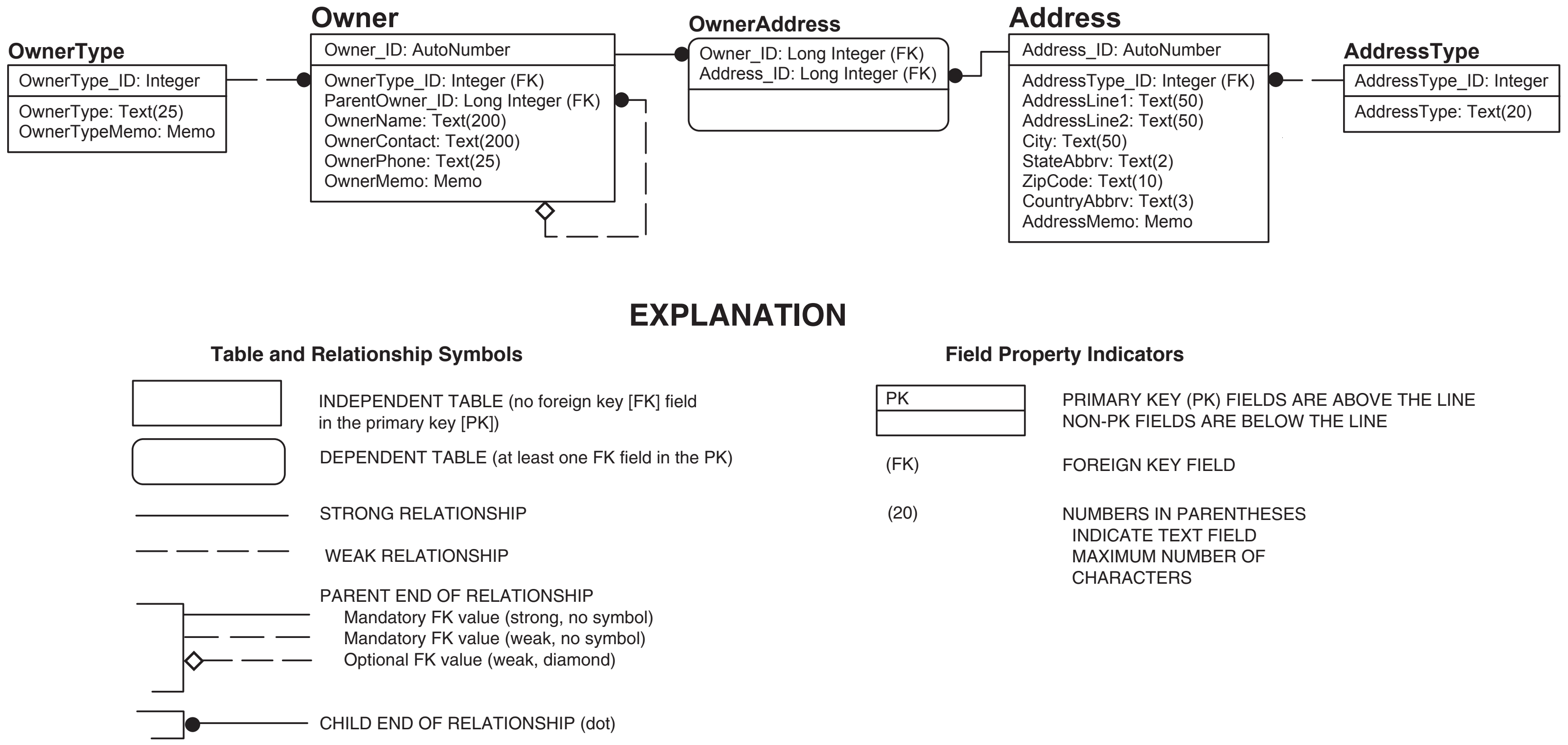

OwnerType
OwnerType_ID: Integer
OwnerType: Text(25)
OwnerTypeMemo: Memo
Owner
Owner_ID: AutoNumber
OwnerType_ID: Integer (FK)
ParentOwner_ID: Long Integer (FK)
OwnerName: Text(200)
OwnerContact: Text(200)
OwnerPhone: Text(25)
OwnerMemo: Memo
OwnerAddress
Owner_ID: Long Integer (FK)
Address_ID: Long Integer (FK)
Address
Address_ID: AutoNumber
AddressType_ID: Integer (FK)
AddressLine1: Text(50)
AddressLine2: Text(50)
City: Text(50)
StateAbbrv: Text(2)
ZipCode: Text(10)
CountryAbbrv: Text(3)
AddressMemo: Memo
AddressType
AddressType_ID: Integer
AddressType: Text(20)
EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ARE ABOVE THE LINE
NON-PK FIELDS ARE BELOW THE LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS

**Figure 1.** Graphical representation of entities, attributes, and relationships (E/R diagram).

the line is dashed. Both strong and weak relationships are illustrated in figure 1. To further help visualize table dependencies, the table is shown with rounded corners in a strong relationship (foreign key is part of the primary key), whereas tables that do not have foreign key dependencies in their primary key have squared corners. In figure 1, strong relationships pair Owners with Addresses in the table *OwnerAddress* (the incoming FK fields also are PK fields in the child table). Weak relationships are represented where the *OwnerType* and *AddressType* tables provide descriptive, non-PK attributes to the tables *Owner* and *Address*, respectively.

Each relationship line has a direction and a cardinality. The direction is recognized by symbols at the parent and child ends of the relationship line. The parent entity in the relationship is represented by no marking on the relationship line or a hollow diamond. The child entity in the relationship is shown by a dot. Direction symbols in an E/R diagram make it easier to visualize the pathway of information flow in a data structure and are especially useful for identifying parent/child entities in a PK/FK relationship in which one table provides attributes to another.

Cardinality symbols display how each record in one entity relates to a count of records in another entity. For example, in a one-to-one (1:1) relationship, each record in the parent entity has exactly one match in the child entity. A one-to-many (1:n) relationship indicates that each record in the parent entity has one or more matching records in the child entity. Many possible cardinality variants can describe the nature of relationships. In the NEWUDS model, all but two of the relationships are one to zero, one or more (1:0,n), and are recognized as a solid or dashed line with the child-end symbol consisting of a solid dot. This type of relationship, illustrated in figure 1, is used when each parent can have zero, one, or more children, but each child must have a parent (the FK cannot be unassigned or Null). Thus, each OwnerType (for example, Municipality) can serve zero, one, or more Owners, and each Owner must have an assigned OwnerType. The zero part of the relationship cardinality allows an OwnerType record to exist in the domain without actually being used by an Owner until needed and is very useful for filling a table with a list of all permissible values before other data are entered into a database.

The other cardinality symbol shown in figure 1 is the self-join for *Owner*. Each Owner record can point to another Owner as its parent by storing the parents' *Owner_ID* in the *ParentOwner_ID field*. The empty diamond on the origin of the self-join relationship line means that the relationship is optional (FK may be Null); that is, an Owner need not have a designated Parent Owner.

## Normalization

Normalization refers to the process of structuring a database to avoid data redundancy, and to reduce complex data elements into their component parts as separate tables and relationships (Fleming and von Halle, 1989, p. 179). The NEWUDS model is structured to approximately third normal form. This means that each table stores information dependent upon a single key, each datum or descriptive element is only recorded in one location, and keys to those data are carefully dispersed among other tables as surrogates for data in the parent table. Because a normalized design often appears to be rather complex and cumbersome to manipulate, queries are constructed that aggregate data from related tables in a denormalized table view. This facilitates data entry, review, or reporting, while the actual data structure is hidden from the user. (The construction of denormalized views is discussed in the section on Operational Issues and Procedures.)

## Keys and Relationships

Each table in the database is designed to hold a specific, defined, and delimited set of data, where each record in the table (a row, or the particular collection of data it represents) is identified by a unique primary key (PK) value (a number). The PK and its value in one table (the parent) can be used in another table (the child) as a surrogate for all the information it represents in a specific row of the source table. In the relationship, the PK from the parent table is called a foreign key (FK) in the child, or receiving table. The relationship established between tables through the shared PK/FK fields provides a path for the data to be combined into a single view of the records from both tables. With many tables related to one another in various combinations, the relational model provides a way to extract data from any two or more tables in any combination desired, if an unambiguous pathway of relationships exists between them. By limiting the field complexity of the data stored in individual tables, the accuracy of data replicated through the key is controlled when it is viewed or combined with data from other parts of the database.

The MS Access database engine protects PK fields in two ways: (1) fields identified as making up the PK must have a value whenever new rows are added to a table; and (2) MS Access prevents the PK values from being duplicated in rows of the table (each PK value is unique within a table). Most data tables in NEWUDS have a single field that serves as the primary key; the key value itself is an incremental integer to represent each row of data in the table. Exceptions are the association tables described below, where a single key field is not practical or desirable. Thus, information-rich keys (fields that have apparently unique values, such as a code name, and could serve to identify data rows) are not allowed to be PK fields, but are relegated instead to attributes identified by an information-neutral numeric key. The possibility that an information-rich key value could be reassigned in practice and thus corrupt associated data in other tables was reason enough to prohibit their use as PKs in the NEWUDS model.

In some tables, a complex key consisting of two or more fields could serve as the PK (for example, the combination of a town code with a state code); however, the convention used in NEWUDS was to relate tables by providing a single field to serve as a surrogate for any such complex PK (association tables excepted). In essence, the design was standardized to use a single field as a PK even when other information fields, alone or in combination, would serve the same purpose. For NEWUDS, storage of a surrogate does not add significantly to the size or complexity of the database, and the convenience of using a surrogate key was considered more important than dealing with the risk of corruption from carrying multiple, complex keys through relationships between tables.

Relationships between tables using keys are modeled and built into the database. The relationship properties, once defined, are enforced by the MS Access database engine. To protect the integrity of the keys across tables, referential integrity constraints are placed on the keys through the relationship. All relationships in NEWUDS have a set of common settings for protection of the keys and the data they represent.

- All tables have a primary key consisting of one or more fields; each record in each table must be identified by a unique key value.

- No keys are optional; primary and foreign key fields are required to hold a value for each record.

- Changes to the value of a primary key are cascaded to child records; all occurrences of that value in the related foreign key in other tables have their values automatically updated.

- Record deletion is restricted if the primary key value also exists as a foreign key value in another table; MS Access displays a warning when children of the record (holding that foreign key value) are in other tables, and prevents related records from becoming orphaned.

These rules, along with enforcement of the uniqueness of primary keys, provide the basic protection needed to prevent key corruption and maintain referential integrity throughout the database. Relationship rules can be adjusted later as needed after experience with NEWUDS.

## Indices

An index is a separate physical object within a relational database. It stores a sorted list of values from one or more fields, with pointers to individual data records having those values. An index is used by a database engine to rapidly identify the location of specific records in tables based upon the indexed values, and it can significantly increase performance when querying or reporting from a database.

NEWUDS is highly conservative in the general use of indices on fields, and it contains only the mandatory indices for all primary and foreign key fields. The index on foreign key fields facilitates look-up activity without requiring any textual values that the keys represent to be sorted or indexed. Other indices should be considered for any fields that could be used for sorting or grouping NEWUDS data. Because identifying the needed sorting and grouping actions on data within NEWUDS is done by the user, creation of additional indices are left to the individual implementations of the database.

## Domain Tables

NEWUDS makes extensive use of lists for classification or description of items in the database. Lists that serve the main data tables and each other constitute 38 of the 62 tables in NEWUDS (61 percent). These tables are called domain tables because each provides specific information describing a particular subject domain. In practice, domain tables contain the set of permissible values from which actual values are taken. An example of a domain table is *tdsState*, in which each record is for an individual State and the fields provide kinds of information about each State (for example, name, postal abbreviation, code number). A relationship with a domain table can therefore provide many different pieces of information through the single value of the inherited key field. This significantly enhances sorting and grouping options for the related data by providing discrete tables in which such actions can be initiated, and it also facilitates maintenance and periodic evaluation of acceptable values for a subject domain.

The value of the many separate domain tables needs to be emphasized, because it is also possible to store those data as fields in the tables they serve. Normalization rules dictate that data be compartmentalized as much as possible to prevent redundancy in the data, which is achieved in NEWUDS by using discrete domain tables. Domain tables provide considerable power to NEWUDS because they can be extended as needed, by adding to the list and by adding new descriptor fields, without affecting the rest of the database design.

In addition to acting as lookup tables, domains are used to create larger domain structures in NEWUDS through clustering and nesting. A domain cluster consists of two or more domain tables joined together in a base table to establish a valid combination of domain values; the base table can have additional attributes that describe properties of each specific combination. Clusters of domain tables can be used to assemble complex sets of descriptive or hierarchical information from a relatively limited set of choices (individual domains), and the result is carried through to a single key value. A domain cluster example in NEWUDS is the RateUnit entity (described in a later section), where three individual domains are combined in a single table to provide any desired combination of decimal, volume, and time terms in a controlled fashion.

Nesting of domains is implemented in NEWUDS where hierarchies of descriptive terms are maintained as separate lists with enforced parent-child relationships. For example, states, counties and minor civil divisions (MCDs), which roughly correspond to towns, are represented as nested domain tables that serve the Location entity for Sites. In an application, nesting would be implemented as the ability to look up a state to reveal its counties and each county's MCDs, or to look up an MCD to reveal its parent county and state. In a nested domain structure, the table that contains the smallest-scale descriptive term (MCD in the example above) is the one that is linked to a data table. This allows any of the larger-scale levels to be retrieved through the nesting relationships to describe the data object they serve.

## Naming Standards

Naming standards convey important information about an item's identity or contents—or both—with little more than an understanding of the convention used. In the NEWUDS database, naming standards apply to both tables and fields.

### Table- and Field-Name Construction

The four general name-construction standards are listed here with explanations.

1. Names are constructed using whole words (with a few exceptions) to provide as much intuitive meaning as possible.

2. Names that are composed of word phrases have the first letter of each word in the phrase capitalized to assist in reading the name.

   Examples of phrased field names include *ConveyanceDetailLabel*, *ConversionToMillion*, *TimeInterval*, and *TransactionEffectiveDate*. Names that do not use whole words in their composition are constructed using a commonly accepted acronym or abbreviation, specifically MCD (Minor Civil Division of the U.S. Census Bureau), HUC (Hydrologic Unit Code), SIC (Standard Industrial Classification), NAICS (North American Industry Classification System), USGS (U.S. Geological Survey), NE (New England), and MGD (million gallons per day, the common unit used for a water transaction/rate). The only other exceptions to whole-word-use in names are when Abbrv is used in place of Abbreviation, and Det is used in place of Determination to help limit the size of item names without affecting their readability, specifically *StateAbbrv*, *CountryAbbrv*, *RateUnitAbbrv*, and *LocationDetMethod*.

3. Names are constructed of only alphabetic characters, with no numerals, punctuation or other special characters (such as spaces or dashes) allowed, except for singular primary key fields that consist of the base table name with an *_ID* suffix.

   The exception here is an important rule of its own: the primary key of each non-associative table is the table root name with a suffix of *_ID*, such as *Location_ID* as the primary key of the *tblLocation* table. This standard of using an *_ID* suffix for a key field allows distinction between key and non-key (data) fields. The presence of *_ID* keys as FK fields in a table also clearly indicates that one or more non-key fields are available in a related table of the same name.

   There are two special variants for the *_ID* rule. The first case involves two fields that have an *_ID* suffix but are not themselves singular primary keys: *ParentOwner_ID* and *ParentSystem_ID*. Use of these fields allows a nested hierarchy to be represented in a single table (a recursion), where each of these fields may hold a value pointing to the PK (*Owner_ID* or *System_ID*, respectively) of another record in the same table. The leading word *Parent* in the field name identifies these recursion keys. In the second case, two fields use a combined name form, *SiteTypeFromID* and *SiteTypeToID*, by pointing to key fields in a domain (*SiteType_ID*) but without a direct, defined relationship between the tables. These fields are used as data rather than keys (PK or FK), and thus must lack the underscore. The table maintenance function of those semi-key data fields is discussed in the section on Operational Issues and Procedures.

4. Fields with a Yes/No data type start with the word "Is," those with a DateTime data type end with the word "Date," and those with a Memo data type end with the word "Memo."

   Fields with a Yes/No (Boolean), DateTime, or Memo data type are unlikely to need modification in a working implementation of the NEWUDS database, so coding the data type into the name provides information about the nature of the field. Coding the data types of all other fields with a suffix or prefix was not considered practical because the data types or sizes of some fields may need to be changed as experience grows with using the database. For example, a Text field may need to be converted to a Memo field if needed storage exceeds 256 characters, or a single precision number may need to be converted to double precision to accommodate unanticipated but valid values for a field.

   Four Yes/No fields exist in NEWUDS: *IsDefaultRate*, *IsNumericDetail*, *IsPrimaryHUC* and *IsPrimaryStateBasin*. They are named as a question phrase to emphasize the Yes/No nature of the value they hold for each data record.

   Four DateTime fields exist in NEWUDS: *SiteDetailEffectiveDate*, *SiteDetailEndingDate*, *TransactionEffectiveDate*, and *TransactionEndingDate*. Each of these fields is used to set a bounding date on a data record.

   There are 28 Memo fields in NEWUDS and all but one use the word "Memo" as a suffix. Memo fields are optional, do not have size limitations, and are used to store remarks or notes about a record in a table. The one Memo field that does not use the Memo term suffix is *SiteContact*. The table *tblSite* contains a *SiteMemo* field for general notes about a Site, whereas the *SiteContact* field stores variable amounts of contact information for a Site and cannot be limited to the 256 characters of a MS Access text field.

### Table Functional Prefixes

For MS Access databases, table prefixes in wide use today are either *tbl* or *tlkp*, where the latter refers to lookup (domain) tables and the former to all other tables. NEWUDS has 62 individual tables, and more than 2 functional table types are recognized. To facilitate rapid identification and communication of the function of different tables in the database, five functionally distinct types of tables are defined by a three-letter prefix applied to the table name (fig. 2). Table names are therefore distinguishable from field names because all table names have lower case prefixes and fields do not. In addition, each of the five table types is displayed using a different color in diagrams of the database structure, imparting significant and immediate understanding of the functional relationships of data elements and relational structures. The five table name prefixes used in NEWUDS and their definitions are provided below.

> *tbl*—basic data table; for example, *tblConveyance*. Basic data tables hold primary data entered into the database and include non-associative core entities. These tables may also function as potential keystone tables for linking to outside databases. The *tbl* tables have a single PK field (*_ID* suffix) that is an integer automatically incremented for each new record (an AutoNumber field) and are guaranteed to be unique by MS Access. These tables always contain one or more fields that are foreign keys (FKs) from domain tables, and thus *tbl* tables inherit classification and other data elements as their own extended attributes. The *_ID* keys from *tbl* tables are dispersed as FK fields among other tables in the database. The nine *tbl* tables in NEWUDS (14 percent) include the following: *tblAddress*, *tblAlias*, *tblConveyance*, *tblLocation*, *tblOwner*, *tblRate*, *tblResource*, *tblSite*, and *tblTransaction*. These *tbl* tables are shown as yellow rectangles in the figures in this report.

The next two functional table types deal with domains, and have a prefix beginning with the letters *td*. These tables are used to provide list information for selective use in other tables. Domain tables are also sometimes known as lookup or reference tables, and naming conventions in use elsewhere may use the *tlkp* prefix.

> *tds*—domain table, static; for example, *tdsState*. Static domain tables hold a list of classification or descriptive items that are used by other tables. The list of items is considered static because it is pre-populated and expected to be complete (for example, the *tdsState* table is fully populated with records describing all the states in New England). The *tds* tables have a single PK field (*_ID* suffix) that is an integer; to prevent inadvertent additions to the static list, the key value must be manually entered whenever new entries are made. Manual key entry also allows creation of custom unique key values for use with the choice list, rather than being limited to automatically generated numbers (for example, adding a "0" key value as a default selection for Unknown). The *tds* tables typically service one or two receiving tables; therefore, *_ID* keys from *tds* tables are typically found in others as a FK. The 24 *tds* tables in NEWUDS (39 percent) include the following: *tdsAddressType*, *tdsConveyanceAction*, *tdsConveyanceActionCategory*, *tdsConveyanceType*, *tdsCounty*, *tdsHUC*, *tdsLocationScale*, *tdsMCD*, *tdsNAICS*, *tdsNEUseType*, *tdsOwnerType*, *tdsRateUnitDecimal*, *tdsRateUnitTime*, *tdsRateUnitVolume*, *tdsResourceType*, *tdsSIC*, *tdsSiteType*, *tdsSiteTypeCategory*, *tdsSiteTypeSubcategory*, *tdsState*, *tdsStateBasin*, *tdsTimeInterval*, *tdsUSGSUseType*, and *tdsWaterBodyType*. These *tds* tables are shown as blue rectangles in the figures in this report.

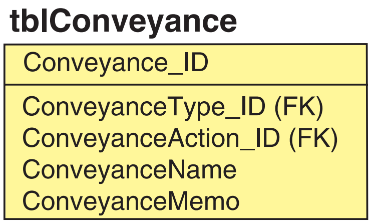
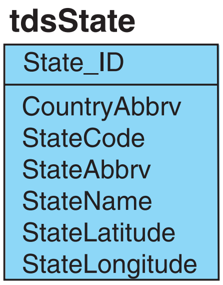
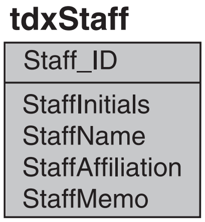
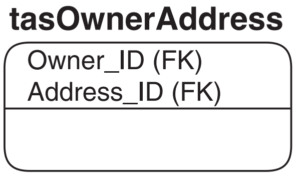
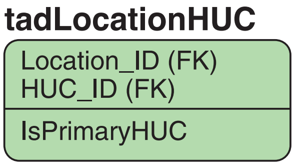
| Functional table prefix | Diagram display format |
|---|---|
| ***tbl* — basic data table**<br>yellow display color<br>rectangle |  **tblConveyance**<br>Conveyance_ID<br>ConveyanceType_ID (FK)<br>ConveyanceAction_ID (FK)<br>ConveyanceName<br>ConveyanceMemo |
| ***tds* — domain, static**<br>blue display color<br>rectangle |  **tdsState**<br>State_ID<br>CountryAbbrv<br>StateCode<br>StateAbbrv<br>StateName<br>StateLatitude<br>StateLongitude |
| ***tdx* — domain, user-extendable**<br>gray display color<br>rectangle |  **tdxStaff**<br>Staff_ID<br>StaffInitials<br>StaffName<br>StaffAffiliation<br>StaffMemo |
| ***tas* — association, simple**<br>white display color<br>rectangle with rounded corners |  **tasOwnerAddress**<br>Owner_ID (FK)<br>Address_ID (FK) |
| ***tad* — association, with data**<br>green display color<br>rectangle with rounded corners |  **tadLocationHUC**<br>Location_ID (FK)<br>HUC_ID (FK)<br>IsPrimaryHUC |

**Figure 2.** Examples of tables with functional prefixes and their diagram display formats.

*tdx*—domain table, user-extendable; for example, *tdxStaff*. User-extendable domain tables provide a list that the user can add to as needed. These tables, therefore, differ in an important way from *tds* tables. The *tdx* tables are considered incomplete at the outset, and cannot be populated fully before the database is used. For example, the *tdxStaff* table will have new entries whenever new staff members need to be identified in the database. The *tdx* tables have a single PK field (*_ID* suffix) that is an AutoNumber field (automatically incremented integer) to facilitate additional entries without regard to key value assignment. Although similar in construction to *tbl* tables, *tdx* tables provide an extendable domain list for characterizing records in data tables, rather than holding primary data for the database. The 14 *tdx* tables in NEWUDS (23 percent) include the following: *tdxAliasLabel*, *tdxConveyanceDetailLabel*, *tdxDataSource*, *tdxLocationDetMethod*, *tdxRateDetailLabel*, *tdxRateMethod*, *tdxRateMethodCategory*, *tdxRateUnit*, *tdxResourceDetailLabel*, *tdxSiteDetailCategory*, *tdxSiteDetailLabel*, *tdxStaff*, *tdxSystem*, and *tdxSystemType*. These *tdx* tables are shown as gray rectangles in the figures in this report.

The next two functional table types are associative, and have a prefix beginning with the letters *ta*. These tables are used to resolve potential many-to-many relationships between two or more tables, or to describe a property relating to a specific pairing of records.

*tas*—association table, simple; for example, *tasOwnerAddress*. Simple association tables are used to resolve many-to-many relationships between two or more tables and are composed of only key fields. In the example *tasOwnerAddress*, constructing an association table allows an Owner to have more than one Address, and a single Address can serve more than one Owner. The association table name is a phrased combination of the two or more parent table names that share the association (in this case, records from tables *tblOwner* and *tblAddress* are paired in the table *tasOwnerAddress*). The *tas* tables have a complex PK composed solely of the FKs of each table involved in the association (in this case, *Owner_ID* and *Address_ID*); no other fields are present in *tas* tables. MS Access maintains the integrity of the association by allowing each combination only once, thus ensuring the primacy of the key. The PKs in association tables can be entered directly to limit a list to valid combinations, or can be assembled as needed by deliberately pairing records from the parent tables in the desired combinations. The seven *tas* tables in NEWUDS (11 percent) include the following: *tasConveyanceAlias*, *tasConveyanceOwner*, *tasOwnerAddress*, *tasResourceAlias*, *tasSiteAddress*, *tasSiteAlias*, and *tasSystemSite*. These *tas* tables are shown as a white rectangle with rounded edges in the figures in this report.

*tad*—association table with data; for example, *tadLocationHUC*. Association tables with data are similar to *tas* tables in that they pair records from two or more tables, but they also store data that describe some property unique to the association itself (that is, to the combination of values forming the PK), or that restricts the pairing in some way. For example, *tadLocationHUC* combines a *Location_ID* with a *HUC_ID* (identifying a specific hydrologic unit or watershed) to form its PK; like *tas* association tables, this allows individual Locations (for example, a county-level aggregate location) to be associated with more than one HUC, and each HUC to be associated with more than one Location. Unlike a *tas* table, however, the *tadLocationHUC* table also adds a Boolean non-key field, *IsPrimaryHUC*, that defines whether an individual pairing of key values represents the primary HUC that should be retrieved with the Location (a Yes value in the field) when more than one HUC is associated. Alternatively, in *tad* tables a data field may be part of the primary key to provide an important constraint on the association (for example, in the table *tadSiteConveyance* discussed in a later section). Thus, *tad* tables resolve potential many-to-many relationships and store data or a constraint unique to a specific association of items from different tables. The eight *tad* tables in NEWUDS (13 percent) include the following: *tadConveyanceDetail*, *tadLocationHUC*, *tadLocationStateBasin*, *tadRateDetail*, *tadResourceDetail*, *tadSiteConveyance*, *tadSiteDetail*, and *tadSiteResource*. These *tad* tables are shown as a green rectangle with rounded edges in the figures in this report.

Recognizing and understanding the naming conventions described above can be a useful aid to exploring the NEWUDS database. Creation of new fields or accessory tables to extend the functionality of the database should use the naming conventions to ensure uniform communication about the identity and functional properties of the new items. Refer to the section of this document on customizing NEWUDS for more information on how this can be accomplished.

## THE NEWUDS DATA MODEL

The full structure of the NEWUDS logical model and physical design is presented in this section. First, the core entities and their relationships are defined and described. The model is then broken into subject areas centered on specific entities, detailed explanations are provided, and new related entities are introduced at the submodel scale. The general use of addresses, data sources, aliases, and user-defined details for different subject areas is discussed in the last four parts of this section.

NEWUDS contains 62 tables, 74 defined relationships, and a total of 256 fields, including 47 unique primary keys dispersed among related tables and 137 non-key data fields. Thirty-eight of the 62 tables in the NEWUDS database are domain tables (62 percent), 15 are association tables (24 percent), and 9 are basic data tables (14 percent). The distribution of non-key fields in tables is similar: of 137 non-key (data) fields in the database, 89 (65 percent) are in the domain tables (including 14 Memo fields for notes about domain entries). The extensive use of domain tables provides flexible classification of the data elements, and the tables serve as the primary objects for grouping, sorting, and summarizing information.

Appendixes 1 and 2 provide the complete NEWUDS data dictionary and listings of the values represented by the domain tables, respectively. The data dictionary provides definitions for each table and field in a tabular format. Readers should refer to those pages for additional details not given below.

## Definition of the NEWUDS Core Entities

The core of the NEWUDS data model can be illustrated by the following statements and conceptual diagrams. More details for each entity are given later in this document.

Each object that can be named as a source or target of water movement is called a Site. Sites can be discrete objects such as a well, water-treatment plant, commercial business, or larger objects represented by a single Site name (for example, a town or local distribution system). Sites are categorized by types and have other definable characteristics. Any two Sites that exchange water are joined by a unidirectional Conveyance, which can be an actual pipe, conduit, or aqueduct, or a virtual representation of the connection between the Sites. A single Site can serve as the input or output end for multiple Conveyances. Together, all Sites and their Conveyances form a water network.

A Transaction/Rate is a record of a single water-movement estimate reported from or calculated for a specific Conveyance between two Sites over a specified time interval. Each Transaction/Rate is characterized at minimum by a rate value (volume per unit time) for the water exchange, a unit of measurement, the time interval covered by the estimate, and the source and method of the estimate. The full history of Transaction/Rates is stored for all Conveyances and the Sites they connect. A representation of estimated monthly Transaction/Rate values for the Conveyance between a well and a treatment plant, as measured and reported by the treatment plant operator, is shown in figure 3.

Two additional entities complete the core of the data model. The spatial Location of each Site is defined by a scale term (point or specified area) and latitude and longitude coordinates (actual point or centroid); more than one Site can realistically share the same Location. Location information provides a spatial reference and visualization potential for the water network. Each Location also links to a hierarchy of other spatial attributes (for example, town, county, state, basin, hydrologic unit). A spatial representation for an example water network is shown in figure 4, in which a withdrawal well and treatment plant share a single point Location, and two town distribution systems (each an additional Site in the data model) are located at town centroid positions. The reciprocal exchange of water between the two distribution systems is indicated by two unidirectional conveyances.

An Owner controls and maintains Sites and Conveyances and can serve as the source of the Transaction data. Each Site and Conveyance is therefore associated with an Owner. Owner type categories include a person, an organization, or a municipal or government agency. Each Owner can own one or more individual elements, and can itself be part of a larger parent organization. For the example shown in figure 4, the well, treatment

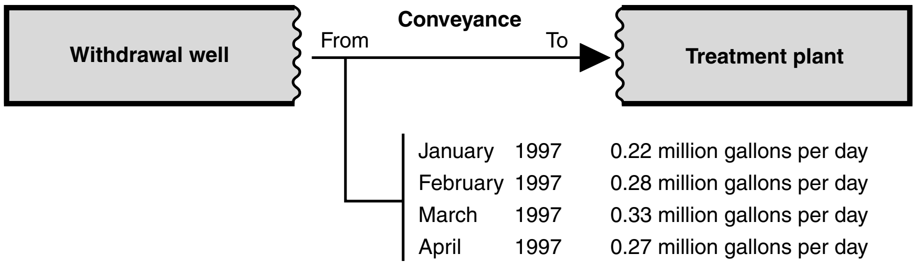

Withdrawal well
Conveyance
From
To
Treatment plant
January 1997 0.22 million gallons per day
February 1997 0.28 million gallons per day
March 1997 0.33 million gallons per day
April 1997 0.27 million gallons per day

**Figure 3**. A representation of a small water network consisting of a withdrawal well and a treatment plant, their unidirectional conveyance, and four estimated monthly transaction/rate values for the conveyance.

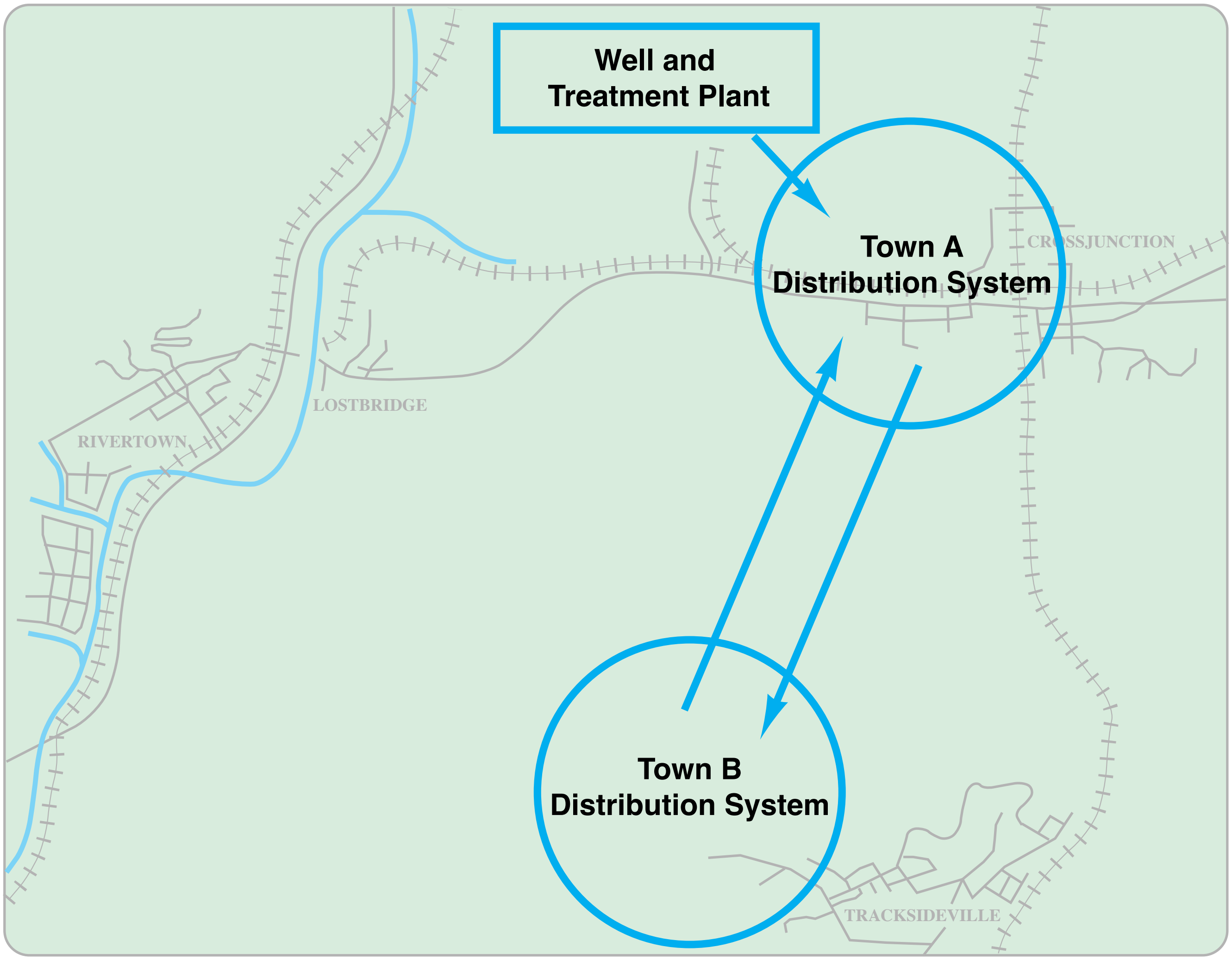


**Figure 4**. Spatial representation of sites (enclosed shapes) and unidirectional conveyances (arrows) making up a small water network.

plant and distribution system A might be owned by Town A, whereas distribution system B could be owned by Town B. Additional information stored about each Owner includes contact and address information.

In summary, pairs of Sites are joined through unidirectional Conveyances, on which are recorded individual water Transaction/Rates. Locations of Sites define the spatial representation of the water network, and information is stored about the Owner of each Site and Conveyance. These statements describe the basic conceptual core of the NEWUDS data model.

## The Core Data Model

The core NEWUDS E/R diagram (fig. 5) is an extension of the conceptual model described previously. (Only the definitions of the entities are shown in figure 5 because the details will be presented in later sections.) The core entities, along with their principal attributes and relationships, are described in the following paragraphs.

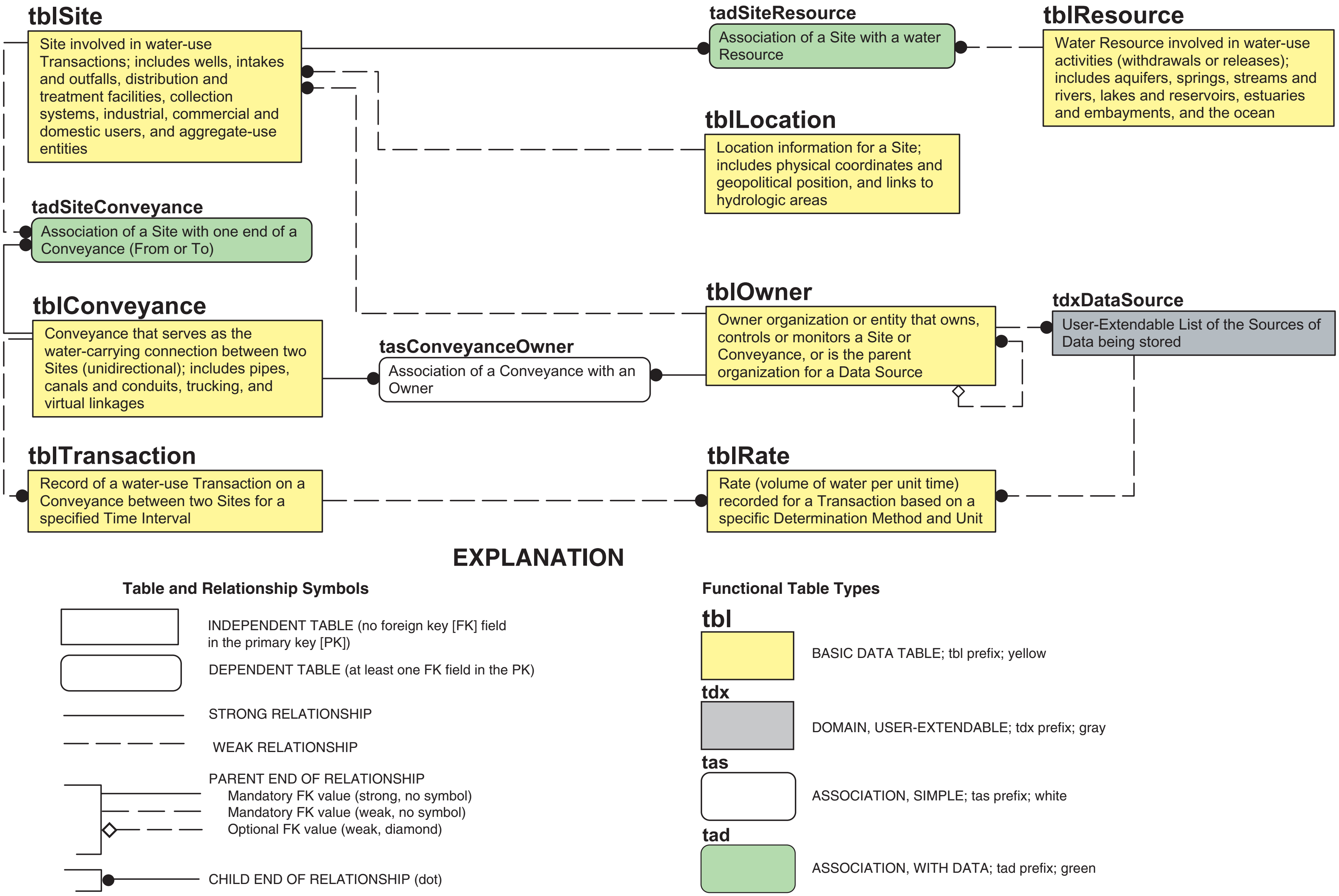

tblSite
Site involved in water-use Transactions; includes wells, intakes and outfalls, distribution and treatment facilities, collection systems, industrial, commercial and domestic users, and aggregate-use entities
tadSiteResource
Association of a Site with a water Resource
tblResource
Water Resource involved in water-use activities (withdrawals or releases); includes aquifers, springs, streams and rivers, lakes and reservoirs, estuaries and embayments, and the ocean
tblLocation
Location information for a Site; includes physical coordinates and geopolitical position, and links to hydrologic areas
tadSiteConveyance
Association of a Site with one end of a Conveyance (From or To)
tblOwner
Owner organization or entity that owns, controls or monitors a Site or Conveyance, or is the parent organization for a Data Source
tdxDataSource
User-Extendable List of the Sources of Data being stored
tblConveyance
Conveyance that serves as the water-carrying connection between two Sites (unidirectional); includes pipes, canals and conduits, trucking, and virtual linkages
tasConveyanceOwner
Association of a Conveyance with an Owner
tblTransaction
Record of a water-use Transaction on a Conveyance between two Sites for a specified Time Interval
tblRate
Rate (volume of water per unit time) recorded for a Transaction based on a specific Determination Method and Unit
EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tas
ASSOCIATION, SIMPLE; tas prefix; white
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 5.** Core tables and relationships in NEWUDS.

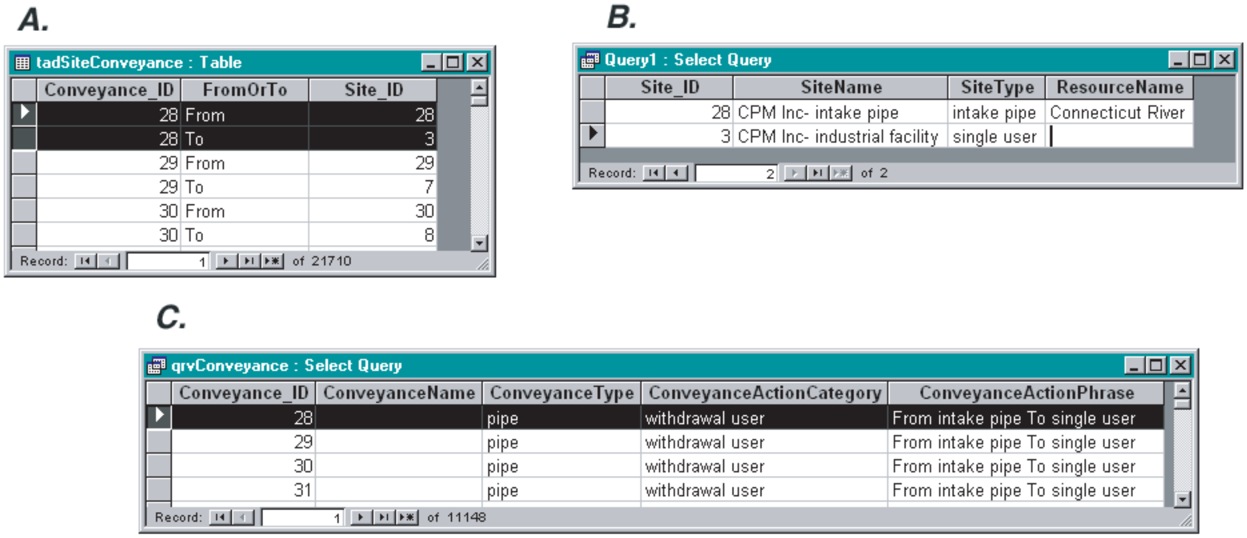
Each object that can be named or identified as the source or target of water movement in the water-use network is called a Site. Sites can be as small as an individual well, or as large as an aggregate of users in a County, if that is the only scale for which actual water-use data are available. Each Site is described by a Location and Owner (or owner organization). A Site that interacts directly with a water resource (withdrawals and returns) is paired with a Resource through an association table, *tadSiteResource*.

All water movements are described as exchanges between individual Site pairs. The connections between these Site pairs for water exchange are called Conveyances, and each unique Conveyance is identified by a *Conveyance_ID*. Conveyances are modeled as strictly unidirectional; therefore, Sites that can exchange water in either direction will have two Conveyances, one for each direction. This is necessary to ensure that all water movements are recorded as non-negative values and to have complete control over defining Sites as either the source or target of individual water Transactions.

The Site-Conveyance relationship is strictly a 2:1 relationship (2 Sites to each Conveyance), and it is the critical element in defining the water-use network. Two Sites form the ends of each Conveyance, one at the input (source) end and the other at the output (target) end. An associative table, *tadSiteConveyance*, is used to pair the two Sites to a single Conveyance. Each Conveyance will have exactly two records in this table, one for each Site to which it joins; this is enforced by the primary key of *tadSiteConveyance,* which consists of the *Conveyance_ID* plus the *FromOrTo* field (which stores the values “From” or “To”). Only one From (source) Site and one To (target) Site can exist for each *Conveyance_ID*. The required non-PK attribute of *tadSiteConveyance* is the *Site_ID* field, which identifies the Site at the From or To Conveyance end. The association table allows multiple Conveyances associated with individual Sites to be quickly identified, and the *FromOrTo* field assists in describing each Conveyance as an input or output connection to a Site. An example of records in the *tadSiteConveyance* table is shown in figure 6, where the highlighted Conveyance identified as *Conveyance_ID* “28” is determined by two queries to be a pipe connecting an intake on the Connecticut River to an industrial facility. Many complex investigations of the data are possible through the association table *tadSiteConveyance*.

The Conveyance attributes include classifications of the type (pipe, aqueduct, and so forth) and the functional action they perform in the water network (such as “From intake pipe To single user” as shown in fig. 6). Each Conveyance may be associated with one or more Owners through the *tasConveyanceOwner* table (fig. 5).

The Transaction entity stores the individual water-movement estimate or measurement made on each Conveyance during a single time interval. Each Transaction in the *tblTransaction* table is uniquely identified by a *Transaction_ID* and has the foreign key *Conveyance_ID* to identify the Conveyance to which the Transaction applies. All Transactions are time-dependent, and a Transaction includes starting and ending dates and the time-interval classification of that date range (for example, year or month). Because different water-use values derived from different estimation methods must be stored separately, original reported rate values are stored in the Rate entity. The *tblRate* table stores the reported/estimated Rate value, along with its original units, method of determination, and data source. One Rate for each Transaction is designated as the default, and its value is converted to million gallons per day equivalents and stored in the *tblTransaction* table for quick retrieval. The Rate entity, therefore, is not needed to retrieve volume



tadSiteConveyance : Table

| Conveyance_ID | FromOrTo | Site_ID |
|---|---|---|
| 28 | From | 28 |
| 28 | To | 3 |
| 29 | From | 29 |
| 29 | To | 7 |
| 30 | From | 30 |
| 30 | To | 8 |

Record: 1 of 21710

Query1 : Select Query

| Site_ID | SiteName | SiteType | ResourceName |
|---|---|---|---|
| 28 | CPM Inc- intake pipe | intake pipe | Connecticut River |
| 3 | CPM Inc- industrial facility | single user | |

Record: 2 of 2

qryConveyance : Select Query

| Conveyance_ID | ConveyanceName | ConveyanceType | ConveyanceActionCategory | ConveyanceActionPhrase |
|---|---|---|---|---|
| 28 | | pipe | withdrawal user | From intake pipe To single user |
| 29 | | pipe | withdrawal user | From intake pipe To single user |
| 30 | | pipe | withdrawal user | From intake pipe To single user |
| 31 | | pipe | withdrawal user | From intake pipe To single user |

Record: 1 of 11148

**Figure 6.** An example of Conveyance data in NEWUDS showing (*A*) records in table *tadSiteConveyance* highlighted for *Conveyance_ID* = "28" and indicating From *Site_ID* = "28" and To *Site_ID* = "3," (*B*) a query showing the identities of the Sites at the Conveyance ends, from an intake pipe to an industrial facility, and (*C*) a query showing the properties of the Conveyance.

estimates for routine summarization but is available for additional details about the original Rates stored for each Transaction. Another attribute of the Transaction/Rate provides identification of the staff person who entered the record.

The last core entity is Owner. An Owner is a person, company, municipality, or other organization that controls and maintains Sites and Conveyances, and can serve as the provider of data (DataSource) for Transaction Rates. Each Owner is uniquely identified by an *Owner_ID*. Individual Owners can be part of an Owner organization, so a *ParentOwner_ID* is identified as an attribute of an Owner record when applicable. This is especially useful for grouping Sites with single Owners or for exploring hierarchies of ownership.

The sections of this document that immediately follow describe and discuss in greater detail the structure of the subject areas of the NEWUDS model. The general use of Addresses, DataSources, Aliases and User-Defined Details are discussed in separate sections.

## Site Subject Area

The Site subject area focuses on the *tblSite* table and its related entities. The Site subject area diagram is shown in figure 7. Each Site is identified by a unique *Site_ID*. Each Site also is strictly classified as a particular SiteType and assigned a water-use functional category. For example, a well Site could be assigned a SiteType (discussed below) value of "withdrawal well" and a use category value of "public-water supply." Sites are paired with *Conveyance_IDs* in the *tadSiteConveyance* table, which connects them (relationally speaking) in a direct path to Conveyance and Transaction information; data from any of these entities are available at the Site (or Site classification) level.

When a Site represents an aggregate covering a geographic area, such as a town rather than an individual Site (point-located), multiple Sites would be entered into NEWUDS to represent different use components for the area. For example, a town aggregate might be represented by separate Sites for ground-water withdrawal, surface-water withdrawal, domestic users, irrigation, and commercial users. This partitioning of an area aggregate into separately identified and classified Site records allows detailed storage of water-use Transactions for the individual components of the aggregate, and it is consistent with storage of point-located Site data. By storing use components as separate Sites in all cases, available data from different scales can be entered, and data summarization is greatly enhanced.

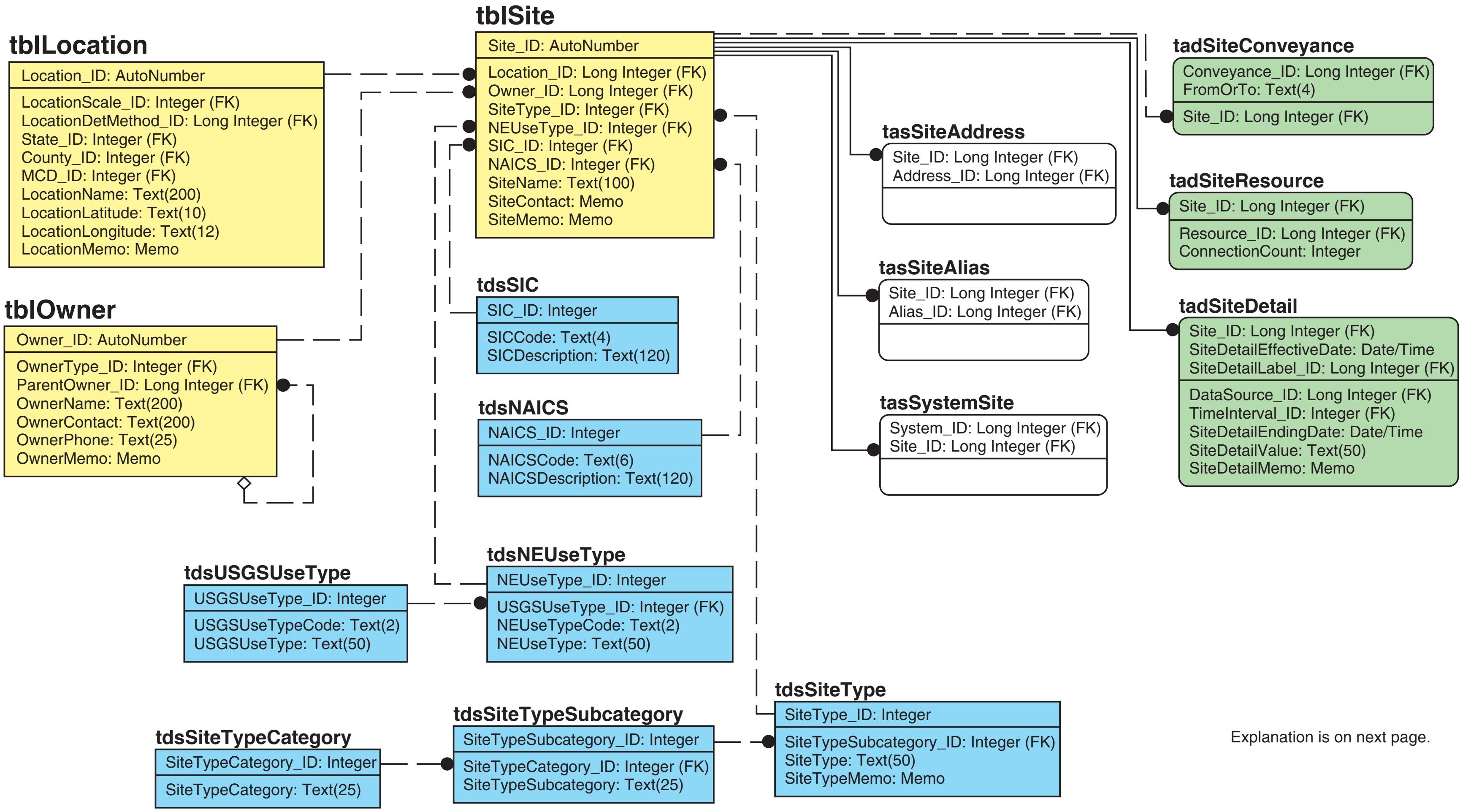
Each Site record contains foreign keys used to associate it with a single Owner and a single Location, which are attributes of the Site. Both entities have numerous descriptive fields of their own that are available for use with the Site object for classification, sorting, and grouping Sites as needed. In particular, the Location represents a data table and 7 domain tables, for a combined total of 30 non-key fields describing spatial, geopolitical, and watershed location information for a Site. The Location and Owner subject areas will be described in separate sections below.

Both the Owner and Location can be shared by different Sites when appropriate. For example, Bottling Company X (the Owner) has a plant facility (Location) with a well and an on-site treatment plant. The well, treatment plant, and facility itself (place of water use for bottling) would be entered and classified as separate Sites in NEWUDS, and all would share the same Location and Owner.

Each Site is given a name but may also be known by other names or through established codes used by different organizations. To accommodate multiple identification methods for single Sites, an associative table, *tasSiteAlias*, is used to pair a *Site_ID* with an *Alias_ID*. Likewise, address information stored for a Site is handled by an association with an Address record in *tasSiteAddress*. Storage of User-Defined Details about a Site is handled through the *tadSiteDetail* table. The Alias, Address, and User-Defined Details subject areas are described later in separate sections.

Each Site is described by a number of domain entities that classify and further identify the attributes or properties of a Site. These include the SiteType, which is a nested domain of category, subcategory and detail (for example, containing the values “treatment,” “preuse,” and “potable treatment plant,” respectively). The type of water use represented by a Site is determined by linkage to the nested domains *tdsUSGSUseType* and *tdsNEUseType* for USGS and New England classifications (for example, “Industrial” and “Textile mill”). Each Site is also assigned U.S. Census Bureau industrial classifications through the *tdsSIC* (Office of Management and Budget, 1987) and *tdsNAICS* (Office of Management and Budget, 1997) domains. Each domain applies a different standardized classification to an individual Site.

Sites that interact directly with a water resource such as a river or reservoir are classified using the *tdsSiteTypeCategory* domain as “resource interactors.” These Sites are associated with the Resource entity through the *tadSiteResource* association table. Each resource-interactor Site is paired with a single Resource, and the number of direct connections represented by the pairing (for example, the number of intake pipes on a reservoir) is stored in the field *ConnectionCount*. Sites also can be associated with other Sites through the *tadSystemSite* association table to form named Systems. The Resource and System subject areas will be discussed separately.


tblLocation
Location_ID: AutoNumber
LocationScale_ID: Integer (FK)
LocationDetMethod_ID: Long Integer (FK)
State_ID: Integer (FK)
County_ID: Integer (FK)
MCD_ID: Integer (FK)
LocationName: Text(200)
LocationLatitude: Text(10)
LocationLongitude: Text(12)
LocationMemo: Memo
tblSite
Site_ID: AutoNumber
Location_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
SiteType_ID: Integer (FK)
NEUseType_ID: Integer (FK)
SIC_ID: Integer (FK)
NAICS_ID: Integer (FK)
SiteName: Text(100)
SiteContact: Memo
SiteMemo: Memo
tadSiteConveyance
Conveyance_ID: Long Integer (FK)
FromOrTo: Text(4)
Site_ID: Long Integer (FK)
tasSiteAddress
Site_ID: Long Integer (FK)
Address_ID: Long Integer (FK)
tadSiteResource
Site_ID: Long Integer (FK)
Resource_ID: Long Integer (FK)
ConnectionCount: Integer
tasSiteAlias
Site_ID: Long Integer (FK)
Alias_ID: Long Integer (FK)
tadSiteDetail
Site_ID: Long Integer (FK)
SiteDetailEffectiveDate: Date/Time
SiteDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
TimeInterval_ID: Integer (FK)
SiteDetailEndingDate: Date/Time
SiteDetailValue: Text(50)
SiteDetailMemo: Memo
tblOwner
Owner_ID: AutoNumber
OwnerType_ID: Integer (FK)
ParentOwner_ID: Long Integer (FK)
OwnerName: Text(200)
OwnerContact: Text(200)
OwnerPhone: Text(25)
OwnerMemo: Memo
tdsSIC
SIC_ID: Integer
SICCode: Text(4)
SICDescription: Text(120)
tdsNAICS
NAICS_ID: Integer
NAICSCode: Text(6)
NAICSDescription: Text(120)
tasSystemSite
System_ID: Long Integer (FK)
Site_ID: Long Integer (FK)
tdsUSGSUseType
USGSUseType_ID: Integer
USGSUseTypeCode: Text(2)
USGSUseType: Text(50)
tdsNEUseType
NEUseType_ID: Integer
USGSUseType_ID: Integer (FK)
NEUseTypeCode: Text(2)
NEUseType: Text(50)
tdsSiteType
SiteType_ID: Integer
SiteTypeSubcategory_ID: Integer (FK)
SiteType: Text(50)
SiteTypeMemo: Memo
tdsSiteTypeSubcategory
SiteTypeSubcategory_ID: Integer
SiteTypeCategory_ID: Integer (FK)
SiteTypeSubcategory: Text(25)
tdsSiteTypeCategory
SiteTypeCategory_ID: Integer
SiteTypeCategory: Text(25)

Explanation is on next page.

**Figure 7.** Site subject area tables, fields, and relationships.

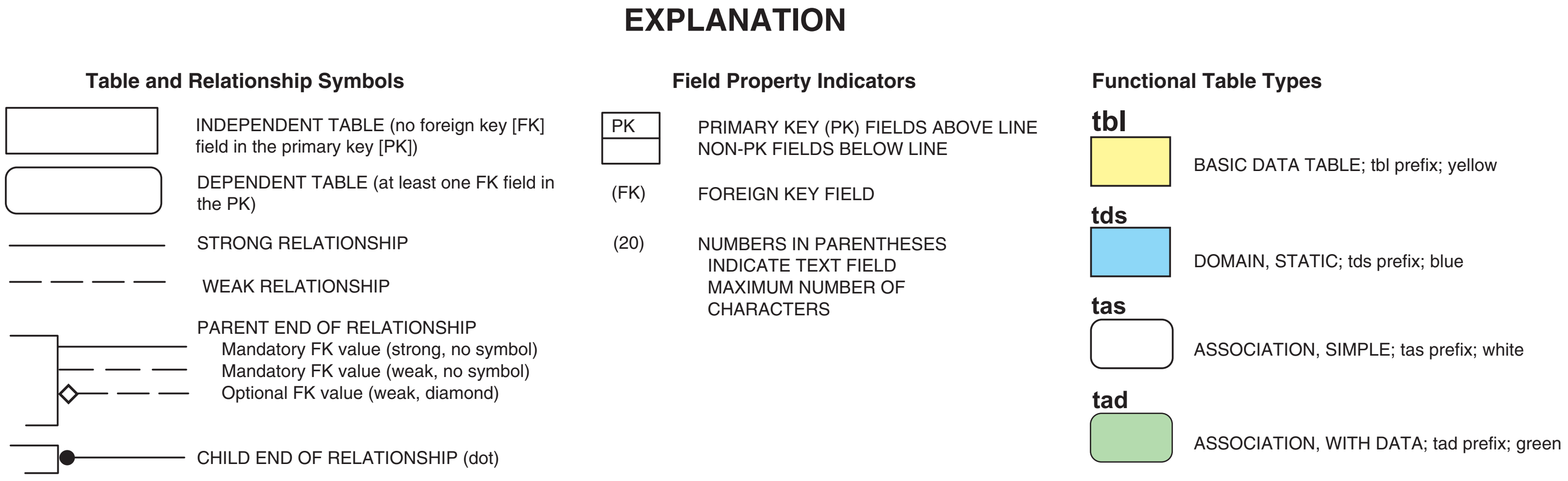

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tas
ASSOCIATION, SIMPLE; tas prefix; white
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 7.** Site subject area tables, fields, and relationships—*Continued.*

Although each user implementation of NEWUDS will create its own Site entries, three generalized water-accounting Sites are provided for users in the *tblSite* table. The first Site has a *SiteName* of “Atmosphere (consumptive use)” and represents water that is evaporated or incorporated into products. To use the Atmosphere Site to capture consumptive-use Transactions for any other Site, a Conveyance is established from the other Site to the “Atmosphere” Site; all Transactions on that Conveyance will represent consumptive use for the other Site. The second generalized Site has a *SiteName* of “Unaccounted-for water” and represents the combination of leakage from distribution systems and public use of water, such as hydrant flushing, fire fighting, and street sweeping. This Site is used as the receiving (To) end of a Conveyance to capture Transaction volumes for those purposes from another Site (for example, to record unaccounted-for use for a regional or local distribution system Site). The third generalized Site has a *SiteName* of “Inflow and Infiltration water” and represents the combination of inflow from surface water and infiltration from ground water into a wastewater-collection system. This Site is used as the sending (From) end of a Conveyance to capture Transaction volumes as inputs to another Site. These three generalized Sites are used at either the From or To end of a Conveyance as appropriate, and are convenient for identifying and gathering together Transactions that they represent (consumptive use, unaccounted-for use, and inflow/infiltration).

## Conveyance Subject Area

The Conveyance subject area focuses on the *tblConveyance* table and its related entities. The Conveyance subject area diagram is shown in figure 8. Each Conveyance is identified by a unique *Conveyance _ID*. Sites are paired with *Conveyance_IDs* in the *tadSiteConveyance* associative table, and Transactions inherit *Conveyance_ID* as an attribute. There is a direct path, therefore, connecting Sites to Transactions through the Conveyance; data from any of these entities are available at the Conveyance level in the database model.

A Conveyance initially is defined as a record in the *tblConveyance* table, but it needs established endpoints in the *tadSiteConveyance* associative table as two paired Sites, one being identified as the source (From end) and the other as the target (To end) using the *FromOrTo* field. This results in two entries in the *tadSiteConveyance* table for each Conveyance. Conveyances are strictly unidirectional; therefore, two Conveyances are required to store data for bi-directional Transactions between two specific Sites (four records in the *tadSiteConveyance* table). Once established, a Conveyance serves as the parent to one or more water Transactions.

To handle consumptive water-use, a Site exists in the *tblSite* table with the name and type classification of “Atmosphere (consumptive use).” Because all Transactions are Conveyance-based in NEWUDS, this special Site is used as the target end of a Conveyance where a Transaction is needed for Site-based consumptive use of water. All consumptive use is, therefore, easily recognized by its Conveyance to the Atmosphere Site, and the *tadSiteConveyance* table can be used to quickly gather all Sites involved in consumptive use. Two other general Sites are also provided in the *tblSite* table and were discussed in the previous section: “Unaccounted-for water” and “Inflow and Infiltration water.”

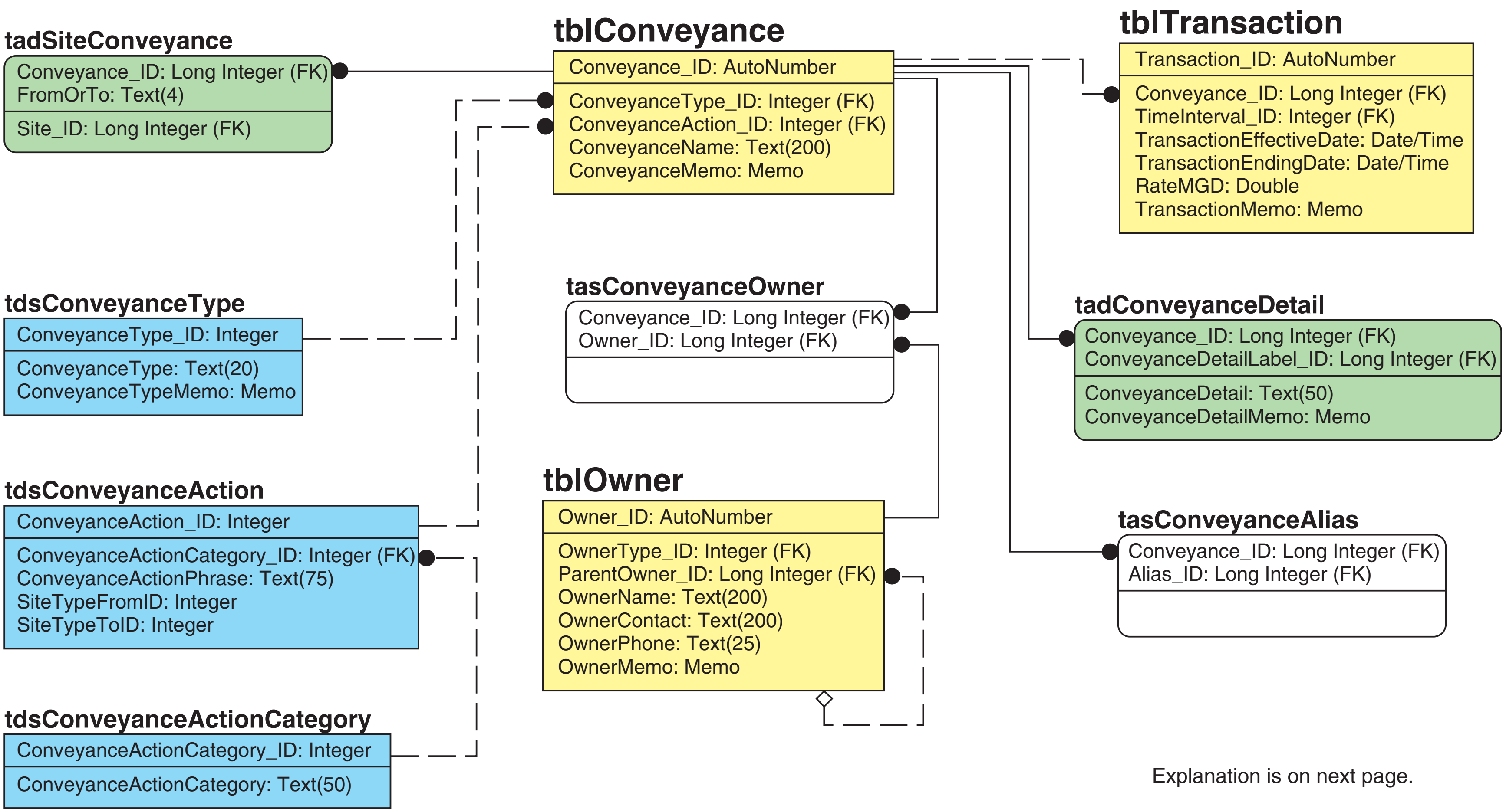
A Conveyance may have one or more Owners responsible for control and maintenance, and this is handled by pairing Conveyances with Owners in the association table *tasConveyanceOwner*. Multiple ownership assignment is useful, for example, when a Conveyance represents water exchange between two towns, and exploring the water network connections to either town (the Owner) would reveal the Conveyance.

Each Conveyance has an optional name (*ConveyanceName*) and is provided access to the *tblAlias* table as needed through the *tasConveyanceAlias* associative table. Storage of User-Defined Details about a Conveyance is handled through the *tadConveyanceDetail* table. The Alias and User-Defined Details subject areas are described elsewhere.

Two domain tables are used to classify each Conveyance: *tdsConveyanceType* and *tdsConveyanceAction*. The *tdsConveyanceType* domain includes the values “pipe,” “canal,” “virtual,” and others. Virtual Conveyances are connections between Sites—such as a connection between a town and a treatment plant—but without detailed data about the Conveyance itself. In this example, the actual number of pipes may not be important or even known, but identifying the connection is critical to capturing Transaction information between the town and treatment plant.

The *tdsConveyanceAction* domain contains a *ConveyanceActionPhrase* field that describes the function of the Conveyance based on the SiteTypes at the source and target ends—for example, “From withdrawal well To potable treatment plant.” The nested domains *tdsConveyanceActionCategory* and *tdsConveyanceAction* can be used to identify Conveyances that serve a particular function within the water-use network, or to contrast Transaction/Rate data between action types, such as comparing ground-water withdrawals sent to treatment plants against ground-water withdrawals sent directly to distribution systems.

Not all SiteTypes can be paired through a connection (for example, a well should not lead directly to wastewater treatment), so the ConveyanceAction domain is static and contains only valid combinations created from the SiteType domain list. The purpose of the static ConveyanceAction domain, therefore, is to limit the possible types of Site connections. Key values for the associated SiteTypes for a particular ConveyanceAction are stored as attributes: *SiteTypeFromID* and *SiteTypeToID*. These pseudo-key fields are not connected directly to the *tdsSiteType* table to avoid a circular relationship in the model (two separate pathways to join tables), but they can be linked through a query whenever needed to extend the ConveyanceAction domain manually to the SiteType domain. The *ConveyanceActionPhrase* field contains values created through a maintenance query (described in a later section) to ensure a semantically accurate phrase defined by the SiteType of the Sites.


tadSiteConveyance
Conveyance_ID: Long Integer (FK)
FromOrTo: Text(4)
Site_ID: Long Integer (FK)
tblConveyance
Conveyance_ID: AutoNumber
ConveyanceType_ID: Integer (FK)
ConveyanceAction_ID: Integer (FK)
ConveyanceName: Text(200)
ConveyanceMemo: Memo
tblTransaction
Transaction_ID: AutoNumber
Conveyance_ID: Long Integer (FK)
TimeInterval_ID: Integer (FK)
TransactionEffectiveDate: Date/Time
TransactionEndingDate: Date/Time
RateMGD: Double
TransactionMemo: Memo
tdsConveyanceType
ConveyanceType_ID: Integer
ConveyanceType: Text(20)
ConveyanceTypeMemo: Memo
tasConveyanceOwner
Conveyance_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
tadConveyanceDetail
Conveyance_ID: Long Integer (FK)
ConveyanceDetailLabel_ID: Long Integer (FK)
ConveyanceDetail: Text(50)
ConveyanceDetailMemo: Memo
tdsConveyanceAction
ConveyanceAction_ID: Integer
ConveyanceActionCategory_ID: Integer (FK)
ConveyanceActionPhrase: Text(75)
SiteTypeFromID: Integer
SiteTypeToID: Integer
tblOwner
Owner_ID: AutoNumber
OwnerType_ID: Integer (FK)
ParentOwner_ID: Long Integer (FK)
OwnerName: Text(200)
OwnerContact: Text(200)
OwnerPhone: Text(25)
OwnerMemo: Memo
tasConveyanceAlias
Conveyance_ID: Long Integer (FK)
Alias_ID: Long Integer (FK)
tdsConveyanceActionCategory
ConveyanceActionCategory_ID: Integer
ConveyanceActionCategory: Text(50)

Explanation is on next page.

**Figure 8.** Conveyance subject area tables, fields, and relationships.

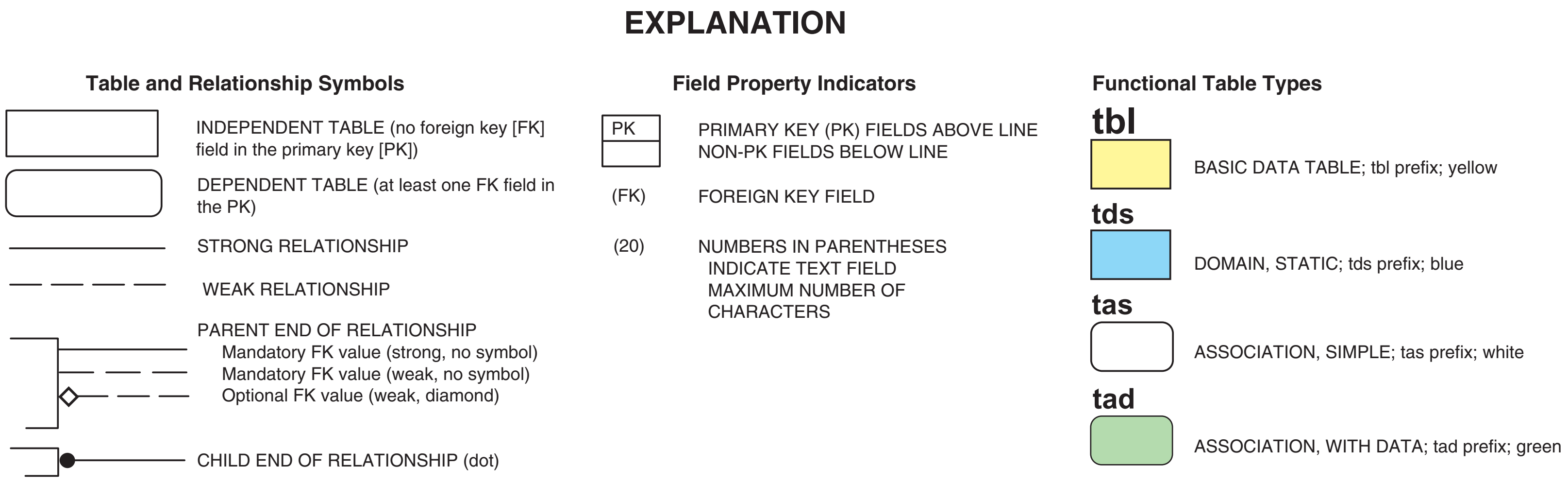

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tas
ASSOCIATION, SIMPLE; tas prefix; white
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 8.** Conveyance subject area tables, fields, and relationships.—*Continued*

## Transaction/Rate Subject Area

The Transaction/Rate subject area focuses on the *tblTransaction* and *tblRate* tables and their related entities. The Transaction/Rate subject area diagram is shown in figure 9. Water Transactions between Sites through Conveyances represent the most detailed data in NEWUDS, and storing the full history of Transactions over time will result in the *tblTransaction* and *tblRate* tables containing more records than other tables in the NEWUDS database. Each Transaction is identified by a unique *Transaction_ID*, and Transactions inherit the *Conveyance_ID* of their parent Conveyance as an attribute of each Transaction. There is, therefore, a direct relational pathway to Site and Conveyance information from the Transaction level of the database model. Details of each Transaction in original reporting units are stored in the table *tblRate*.

Transactions are time dependent. Each Transaction is bounded, inclusively, by its two date fields, *TransactionEffectiveDate* and *TransactionEndingDate*. The interval category that the date range defines is classified by a value from the *tdsTimeInterval* domain table ("year," "month," and so forth). It is assumed that single day data represents the smallest recording interval for Transaction/Rate data. The TimeInterval domain is useful for aggregating data from different time scales and for analyzing the occurrence frequency of different scales for various Conveyance and Site types. Other time scales, fiscal years (Federal and State), water years, and seasons, can be developed through queries on the bounding date fields and are not designed into the database.

NEWUDS specifications require that Rates for a specific Transaction be allowed to vary based on different determination methods. This facilitates study of differences in the determination methods used for the summarization and reporting of water-use activities. Original Rate values (as provided by the DataSource), therefore, are stored along with their descriptive attributes in the table *tblRate*. The relationship between *tblTransaction* and *tblRate* is modeled as one-to-many, with many Rate estimate alternatives possible for each Transaction. When there is more than one Rate record, a field in *tblRate*, *IsDefaultRate*, identifies the particular Rate record to be associated by default with the Transaction; this Rate is converted to common units and stored in the *RateMGD* field in the *tblTransaction* table using a maintenance query described in a later section. When only a single Rate is stored for a Transaction, it is always the default Rate. Having a single default Rate stored in converted form in the *tblTransaction* table also means that for most data exploration and reporting activities, the *tblRate* table and its associated domains are not needed.

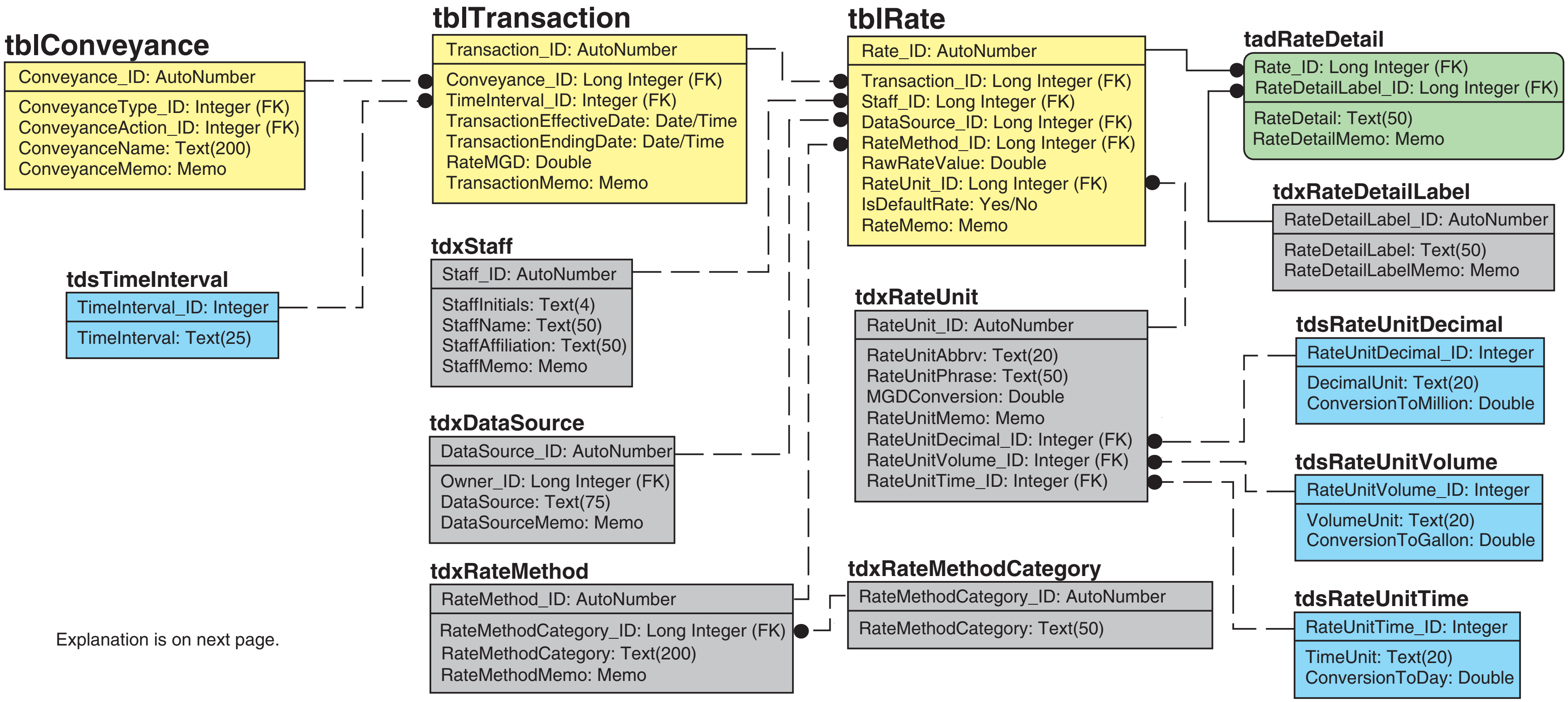
The original data for each Transaction is stored in the *tblRate* table. The *RawRateValue* field stores the numeric part of a reported Rate value. For example, "1.45" would be stored for 1.45 thousand cubic feet per second. It is important to store all Rates in their reported form to facilitate checking against original DataSources. Within the *tblRate* table, the *RawRateValue* is associated with its original RateUnit, its method of determination (nested domain of RateMethod and parent category), a DataSource, and the Staff responsible for the entry. The *tdxRateUnit* domain table represents a domain cluster, allowing a RateUnit to be created from any combination of decimal, volume, and time dimensions used in the original reporting units (for example, Rates reported in thousand cubic feet per second have separate unit component terms of “thousand,” “cubic-feet,” and “second”). Each RateUnit also is provided with a value for the *MGDConversion* field, which is the value to multiply the *RawRateValue* by to convert it to common units regardless of the reporting units. The *RawRateValue* is converted, based on the RateUnit, to an equivalent value in million gallons per day (MGD) units, and that converted Rate value is stored in the *tblTransaction* table in the field *RateMGD*.

During initial data entry or batch loading, *RateMGD* in *tblTransaction* is assigned a default value of “-1” for new Transaction records, and a maintenance query (described in a later section) needs to be run to apply the RateUnit *MGDConversion* against the *RawRateValue* to update the Transaction *RateMGD* value. The conversion factors are developed for any RateUnit combination by automatically creating the product of each component conversion (decimal, volume, and time). When new RateUnits are added to the domain, a maintenance query (described in a later section) is used to update the *RateUnitPhrase* and *MGDConversion* fields in *tdxRateUnit* based on the selection of a unique combination of components. The *tdxRateUnit* table also could be used in reverse, through queries, to convert the commonly used million gallons per day *RateMGD* values to any other units desired.

One additional feature of the Transaction subject area, the RateDetail, is shown in figure 9. User-Defined Detail tables will be discussed in a separate section later in this document. Initial review of the NEWUDS database model led to a request that accuracy information be stored with the Rate estimates. The association of a *Rate_ID* with a *RateDetailLabel_ID* and a *RateDetail* in the *tadRateDetail* table can accommodate any optional attributes for Rates, including accuracy. Accuracy values can be associated with each Rate by assigning a value of “Accuracy” to a record in the *RateDetailLabel* field of the *tdxRateDetailLabel* domain and recording the accuracy estimate in the *RateDetail* field for each *Rate_ID*. In addition, a note about the definition of accuracy for a particular NEWUDS implementation can be entered in the *RateDetailLabelMemo* field.

## Location Subject Area

The Location subject area focuses on the *tblLocation* table and its related entities. The Location subject area diagram is shown in figure 10. Location is modeled as an attribute of a Site. The *Location_ID* as a foreign key in the *tblSite* table provides the full collection of Location attributes and related domains to individual Sites. Through the Site’s relationships, Location fields can be used as attributes that also provide spatial characterization of Conveyances and Transactions. A single Location may accurately serve more than one Site if they occur in the same place, such as individual Site-level objects that represent various water-use types (for example, livestock or irrigation) of an aggregate Site. Site aggregates were discussed previously in the Site Subject Area section.


tblConveyance
Conveyance_ID: AutoNumber
ConveyanceType_ID: Integer (FK)
ConveyanceAction_ID: Integer (FK)
ConveyanceName: Text(200)
ConveyanceMemo: Memo
tblTransaction
Transaction_ID: AutoNumber
Conveyance_ID: Long Integer (FK)
TimeInterval_ID: Integer (FK)
TransactionEffectiveDate: Date/Time
TransactionEndingDate: Date/Time
RateMGD: Double
TransactionMemo: Memo
tblRate
Rate_ID: AutoNumber
Transaction_ID: Long Integer (FK)
Staff_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
RateMethod_ID: Long Integer (FK)
RawRateValue: Double
RateUnit_ID: Long Integer (FK)
IsDefaultRate: Yes/No
RateMemo: Memo
tadRateDetail
Rate_ID: Long Integer (FK)
RateDetailLabel_ID: Long Integer (FK)
RateDetail: Text(50)
RateDetailMemo: Memo
tdxRateDetailLabel
RateDetailLabel_ID: AutoNumber
RateDetailLabel: Text(50)
RateDetailLabelMemo: Memo
tdsTimeInterval
TimeInterval_ID: Integer
TimeInterval: Text(25)
tdxStaff
Staff_ID: AutoNumber
StaffInitials: Text(4)
StaffName: Text(50)
StaffAffiliation: Text(50)
StaffMemo: Memo
tdxRateUnit
RateUnit_ID: AutoNumber
RateUnitAbbrv: Text(20)
RateUnitPhrase: Text(50)
MGDConversion: Double
RateUnitMemo: Memo
RateUnitDecimal_ID: Integer (FK)
RateUnitVolume_ID: Integer (FK)
RateUnitTime_ID: Integer (FK)
tdsRateUnitDecimal
RateUnitDecimal_ID: Integer
DecimalUnit: Text(20)
ConversionToMillion: Double
tdxDataSource
DataSource_ID: AutoNumber
Owner_ID: Long Integer (FK)
DataSource: Text(75)
DataSourceMemo: Memo
tdsRateUnitVolume
RateUnitVolume_ID: Integer
VolumeUnit: Text(20)
ConversionToGallon: Double
tdxRateMethod
RateMethod_ID: AutoNumber
RateMethodCategory_ID: Long Integer (FK)
RateMethodCategory: Text(200)
RateMethodMemo: Memo
tdxRateMethodCategory
RateMethodCategory_ID: AutoNumber
RateMethodCategory: Text(50)
tdsRateUnitTime
RateUnitTime_ID: Integer
TimeUnit: Text(20)
ConversionToDay: Double

Explanation is on next page.

**Figure 9.** Transaction/Rate subject area tables, fields, and relationships.

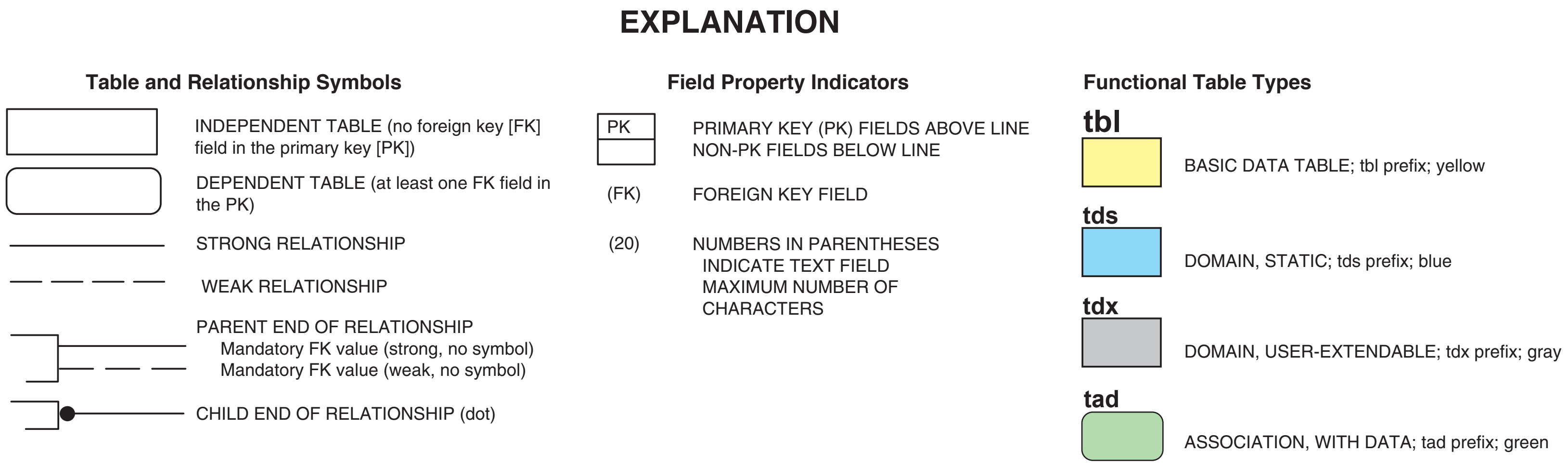
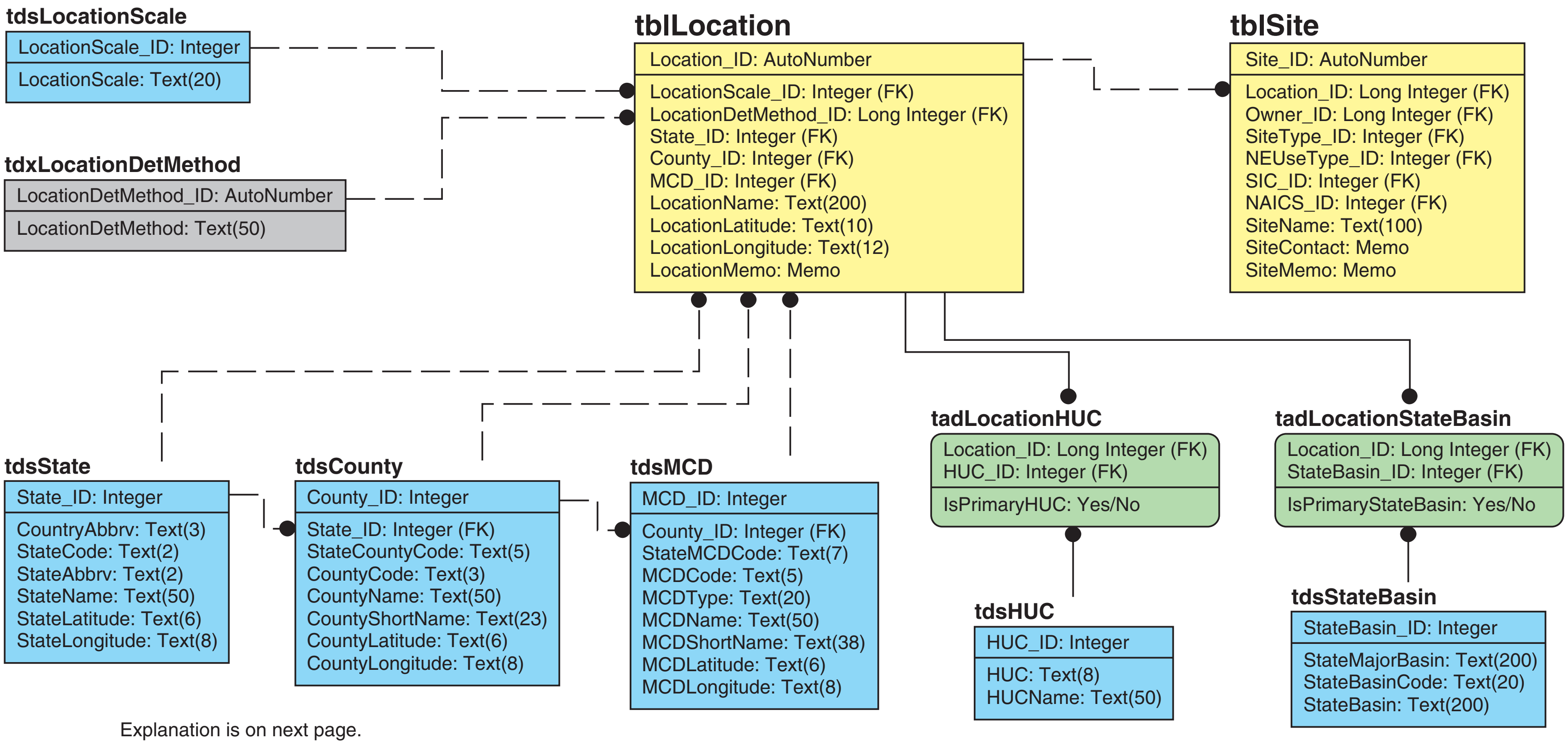

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 9.** Transaction/Rate subject area tables, fields, and relationships—*Continued.*



Explanation is on next page.

**Figure 10.** Location subject area tables, fields, and relationships.

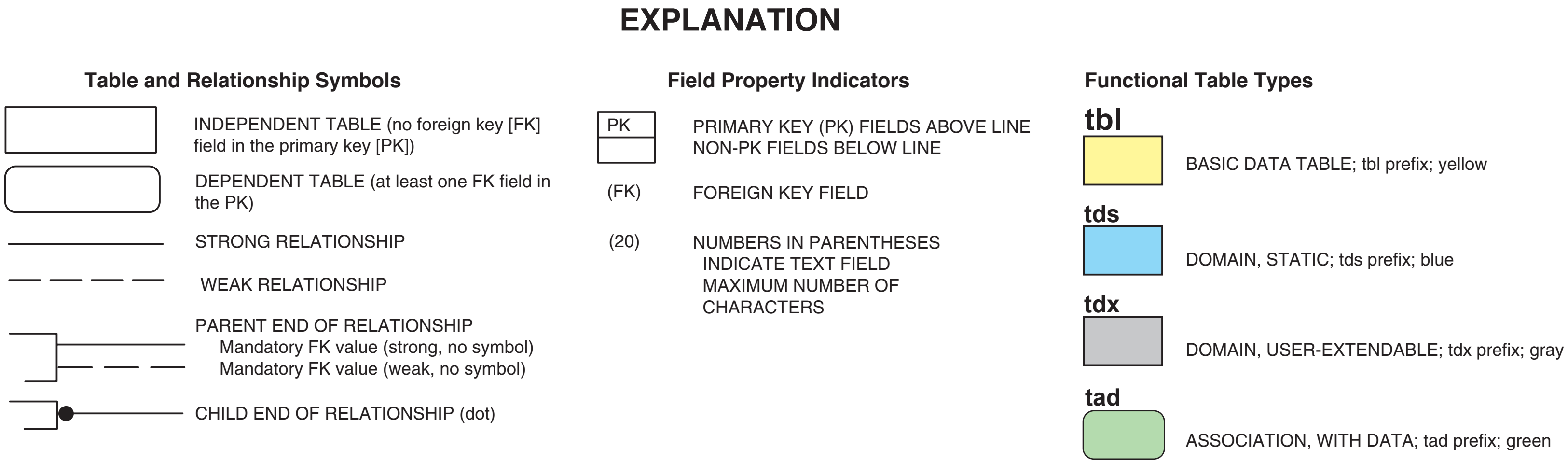

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 10.** Location subject area tables, fields, and relationships—*Continued*.

Each Location is identified by a unique *Location_ID*, with optional fields for a *LocationName* and mapping coordinates. *LocationLatitude* and *LocationLongitude* can store coordinates representing a point location or the centroid of a larger area.

Each Location is characterized by domains representing LocationScale (point, or various area categories), the method used for determining a Location (LocationDetMethod), and a geopolitical state, county, and MCD nested hierarchy. Although nested, the geopolitical domains each have their own foreign key in *tblLocation*, and each domain table has a record coded for not applicable. This allows a Site Location to be defined at a large scale (for example, county data could not have a single valid MCD value) or as an irregular area that does not correspond to geopolitical boundaries, such as for a regional distribution system. For larger-than-MCD area Locations, the *MCD_ID* field in *tblLocation* would use the key value pointing to not applicable; for larger-than-County Locations, both the *County_ID* and *MCD_ID* fields would be coded as not applicable.

Locations also can be associated with one or more standard hydrologic units or State basin classification schemes through the associative tables *tadLocationHUC* and *tadLocationStateBasin*. The *tdsHUC* domain contains a national 8-digit watershed coding scheme, whereas the *tdsStateBasin* domain represents a place to store an individualized state watershed coding scheme. Because a single area Location may cross basin boundaries, more than one HUC or StateBasin may be associated with it, requiring one to be designated as the primary selection from the domain to be used by default. This is accomplished by using the *IsPrimaryHUC* and *IsPrimaryStateBasin* fields in the association tables; for example, the record with a value of “Yes” for *IsPrimaryHUC* would be the default HUC for that Location. When only a single HUC or StateBasin is associated with a Location, it is always the default.

## Owner Subject Area

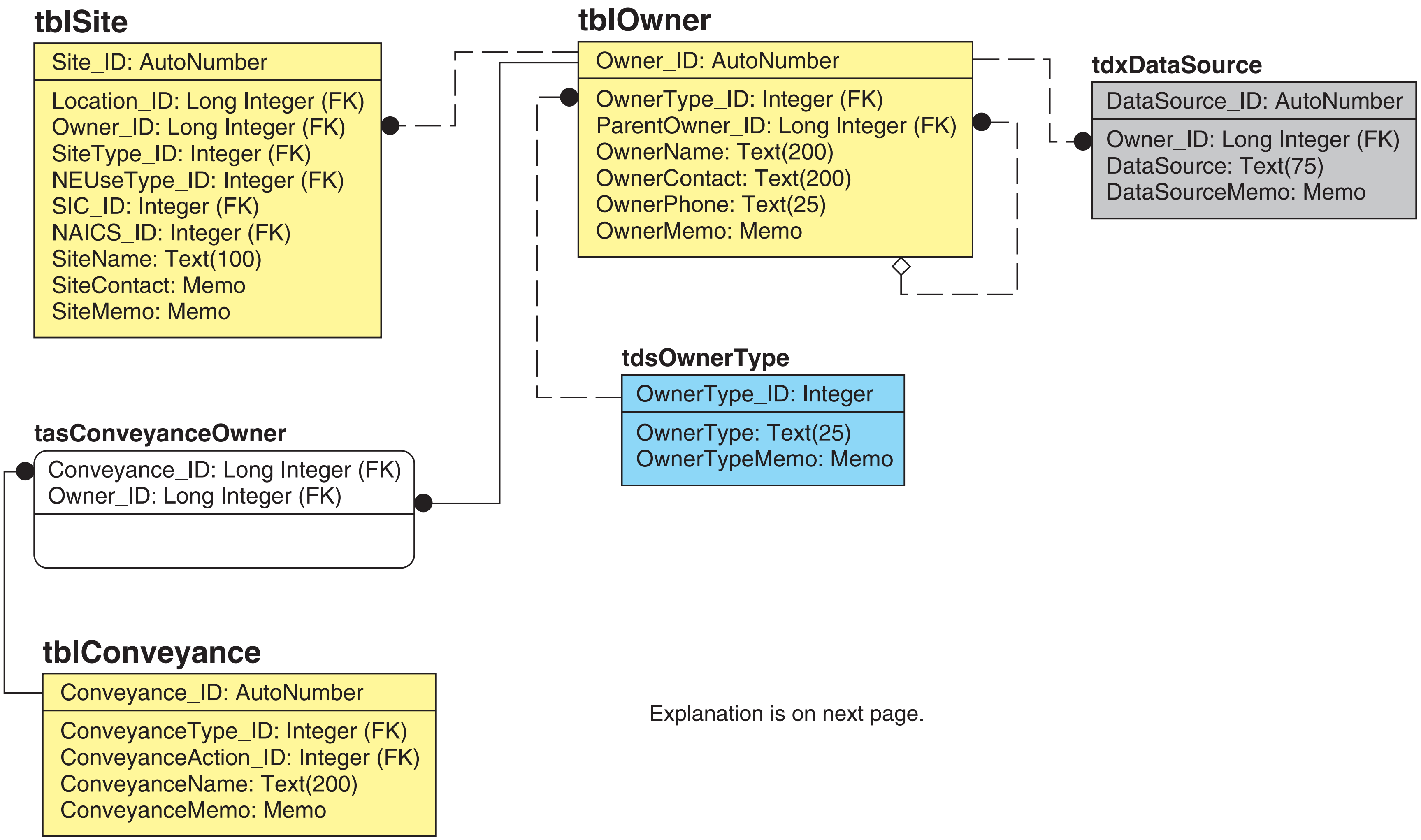
The Owner subject area focuses on the *tblOwner* table and its related entities. The Owner subject area diagram is shown in figure 11.  An Owner is a person, company, municipality, or other organization that controls and maintains a Site or Conveyance, or that can serve as the provider of a DataSource for Transaction Rates and User-Defined Details in the database. Each Owner is identified by an *Owner_ID*, and aggregate areas represented as single Sites, such as a town, may have no Owner in this context and would be assigned an *Owner_ID* corresponding to a value of “No Owner.” Owners are further described by fields for *OwnerName*, *OwnerContact,* and *OwnerPhone*. Different kinds of Owners are classified through the *tdsOwnerType* domain (for example, “Private,” “Municipal,” or “State”).

Sites and DataSources are associated with a single Owner by inheriting the *Owner_ID* as a FK attribute. Conveyances may have one or more Owners (for example, recognizing joint ownership of a connection between towns), which is accommodated through the *tasConveyanceOwner* association table. Individual Owners can be identified as part of an Owner organization through the hierarchical association of a *ParentOwner_ID* with a different Owner record (a self-join). Allowing nesting of Owners provides a way to group Sites and Conveyances into actual organizational units and to explore hierarchies of ownership.

## Resource Subject Area

The Resource subject area focuses on the *tblResource* table and related entities. The Resource subject area diagram is shown in figure 12. The NEWUDS model incorporates a *tblResource* table to store identification information for all aquifers, rivers, streams, reservoirs, lakes, ponds, and estuaries associated with water-use activities. A Site that interacts with a Resource is classified as a “resource interactor” (SiteTypeCategory) and is paired with a single Resource in the *tadSiteResource* associative table. There is an option to record the number of connections represented by the pairing (individual pipes or wells) in the *ConnectionCount* field. From these relationships, all withdrawals and releases on any water body listed in the *tblResource* table are available.

Resources should not be equated with Locations in NEWUDS. Locations serve to identify the places where Sites reside, whereas Resources identify water bodies in the major categories of aquifer, river/stream, and reservoir, among others. Resources are independent of Sites, although those Sites that interact with a water Resource will likely have a Location that matches the position of some part of its associated Resource.



Explanation is on next page.

**Figure 11.** Owner subject area tables, fields, and relationships.

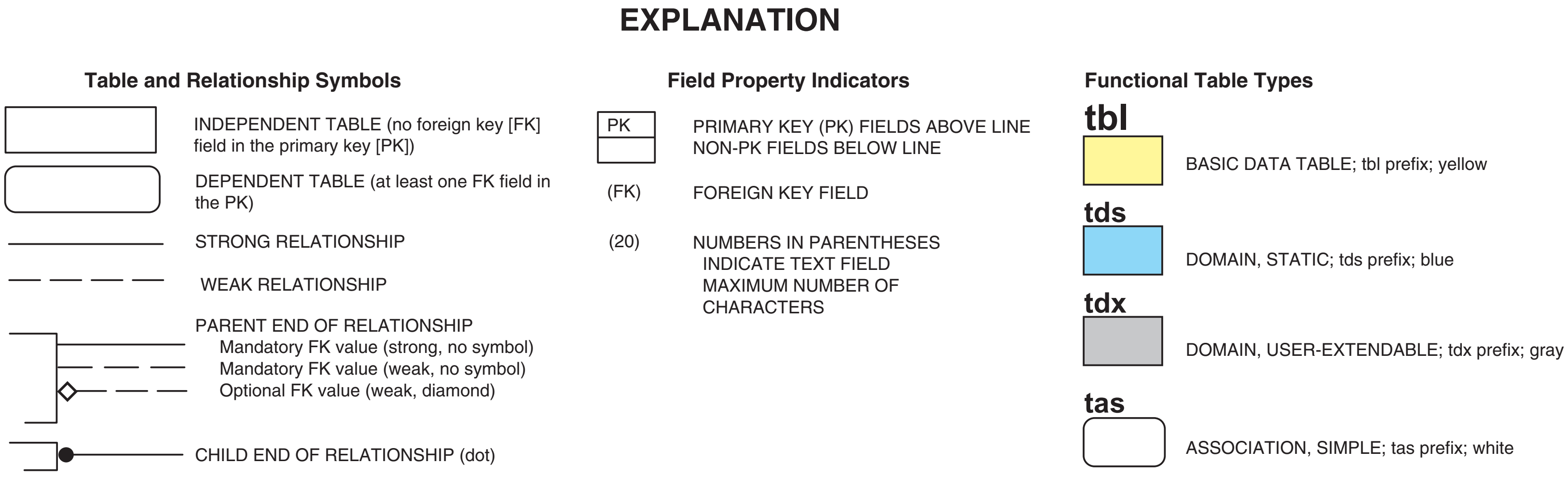
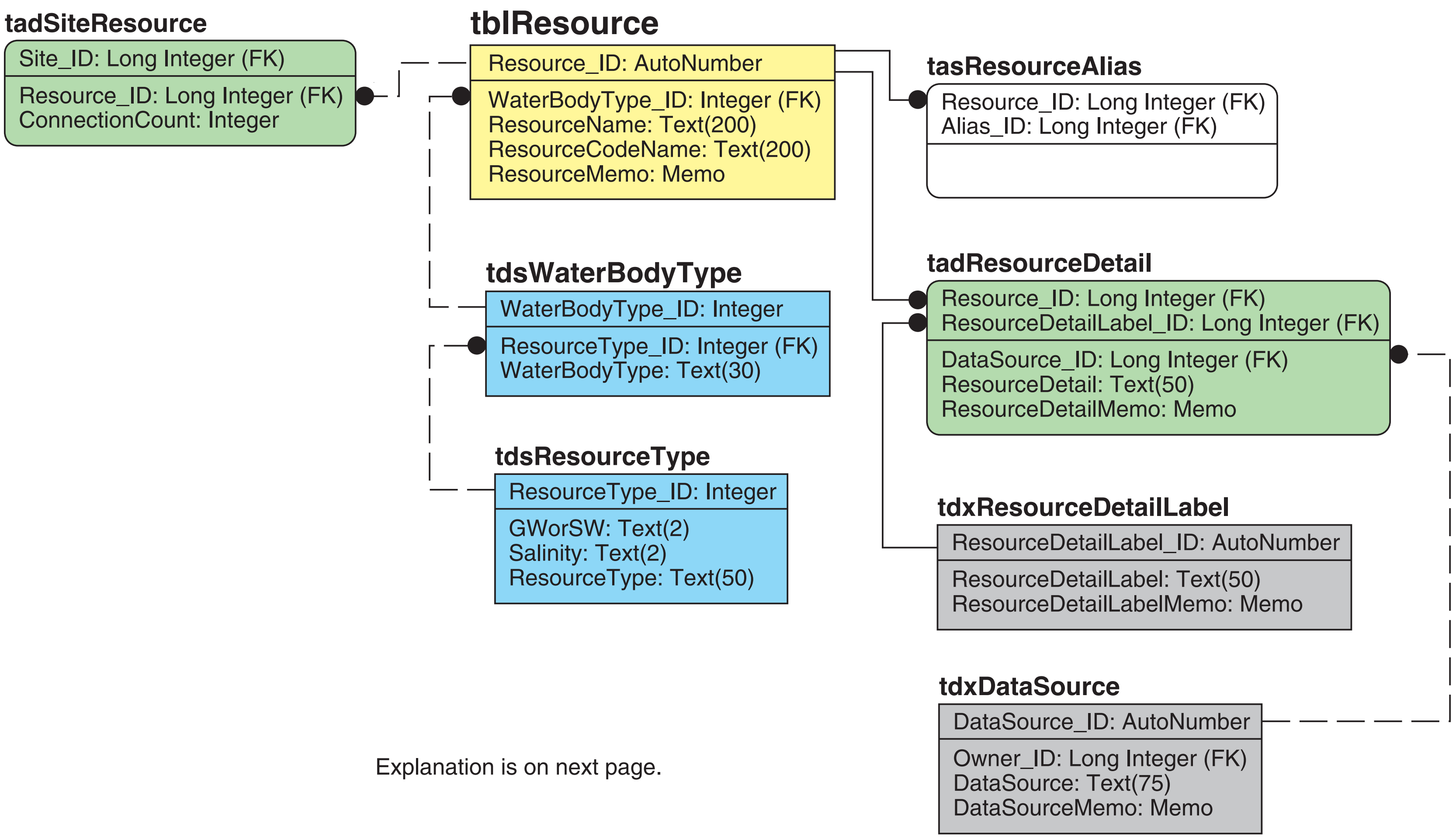

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tas
ASSOCIATION, SIMPLE; tas prefix; white

**Figure 11.** Owner subject area tables, fields, and relationships—*Continued*.


tadSiteResource
Site_ID: Long Integer (FK)
Resource_ID: Long Integer (FK)
ConnectionCount: Integer
tblResource
Resource_ID: AutoNumber
WaterBodyType_ID: Integer (FK)
ResourceName: Text(200)
ResourceCodeName: Text(200)
ResourceMemo: Memo
tasResourceAlias
Resource_ID: Long Integer (FK)
Alias_ID: Long Integer (FK)
tdsWaterBodyType
WaterBodyType_ID: Integer
ResourceType_ID: Integer (FK)
WaterBodyType: Text(30)
tadResourceDetail
Resource_ID: Long Integer (FK)
ResourceDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
ResourceDetail: Text(50)
ResourceDetailMemo: Memo
tdsResourceType
ResourceType_ID: Integer
GWorSW: Text(2)
Salinity: Text(2)
ResourceType: Text(50)
tdxResourceDetailLabel
ResourceDetailLabel_ID: AutoNumber
ResourceDetailLabel: Text(50)
ResourceDetailLabelMemo: Memo
tdxDataSource
DataSource_ID: AutoNumber
Owner_ID: Long Integer (FK)
DataSource: Text(75)
DataSourceMemo: Memo

Explanation is on next page.

**Figure 12.** Resource subject area tables, fields, and relationships.

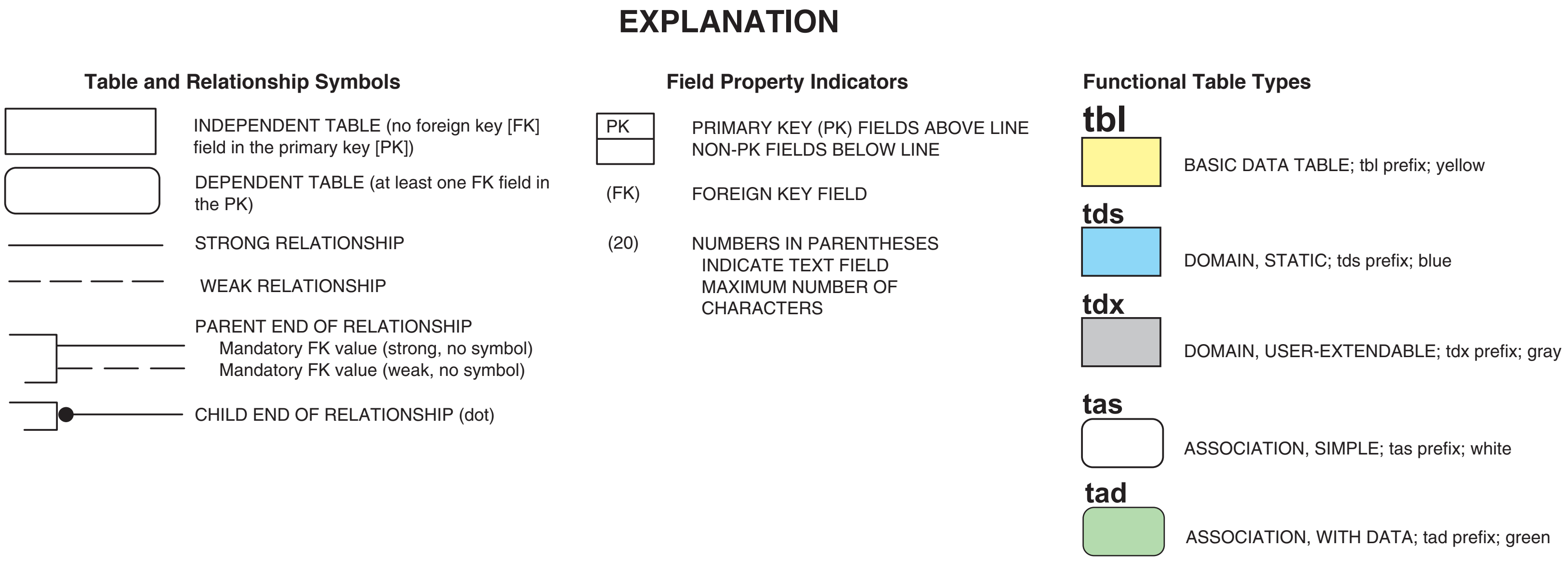

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tas
ASSOCIATION, SIMPLE; tas prefix; white
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 12.** Resource subject area tables, fields, and relationships—*Continued*.

Each Resource is identified by a *Resource_ID*, a mandatory *ResourceName*, and an optional *ResourceCodeName*. Two name fields are used because names of water bodies are commonly redundant, even within the same watershed. The ubiquity of rivers and streams named Mill Brook in New England is a good example. The *ResourceName* field is intended for storage of the name in common use (for example, "Mill Brook"), whereas the *ResourceCodeName* field is for an extended version of the name that provides unique identification information. A standard naming rule for *ResourceCodeName* has not been specified in NEWUDS, but the suggested practice is that different rivers with the same *ResourceName* in the database be distinguished by including the name of the town nearest its source in the *ResourceCodeName*. Thus, the *ResourceCodeNames* of "Mill Brook near Springfield, MA" and "Mill Brook near Concord, MA" are distinguishable, even though both have the same common name (*ResourceName*) of "Mill Brook." That distinction facilitates searches in the database for Sites that interact with only one of those water bodies. Development and implementation of the naming scheme does not affect the current structure of NEWUDS in any way because only the contents of the *ResourceCodeName* field are affected.

Nested ResourceType and WaterBodyType domains provide a classification system for individual Resources and support the needs of a national water-use compilation where, for example, withdrawals associated with surface water are compared to those from ground water. The *tdsResourceType* domain table identifies ground- or surface-water sources and a salinity attribute (freshwater, brackish, or saline), and applies a *ResourceType* name to each combination (such as "surface-water, fresh"). These basic classifications are then assigned as parent categories for individual selections in the *tdsWaterBodyType* table, which includes values such as "Reservoir," "River," and "Estuary." These domains and their relationships in NEWUDS allow the identification of those Sites (and therefore their Conveyances and Transactions) involved with different types of water Resources.

The ability to store optional information about any particular Resource is provided by the *tadResourceDetail* table and its associated *tdxResourceDetailLabel* and *tdxDataSource* tables. This allows dams to be associated with reservoirs, fisheries to be associated with particular rivers, and any other optional attributes to be classified and stored as needed. Resources also may have Aliases stored in the *tblAlias* table through the *tasResourceAlias* associative table. Both the User-Defined Details and Alias subject areas are described elsewhere in this report. To meet potential future needs, the Resource subject area entities and attributes can extend NEWUDS by linking to hydrologic and geopolitical areas, gage information, ecology, and other water-body specific data.

## System Subject Area

The core NEWUDS data structures that join Sites through unidirectional Conveyances describe a water network by allowing each Site to show the characteristics of its input and output connections and related Transactions. Sites can be grouped by their descriptive attributes and Type domains, and by the attributes of their Conveyances, Transactions/Rates, Locations, Owners, and associated Resources. For convenience, an additional grouping of Sites called Systems can be set up in NEWUDS to gather Sites for any purpose, regardless of whether they are connected or share any attributes. For example, a public-supply System could be created to include all Sites that represent withdrawals (wells, intakes), treatment, and distribution. Other Systems could include the cluster of Sites that make up a fish hatchery operation, an industrial complex, the distribution network of Sites for a town, a regional collection system, or the collection of Sites handled as a discrete unit during data entry.

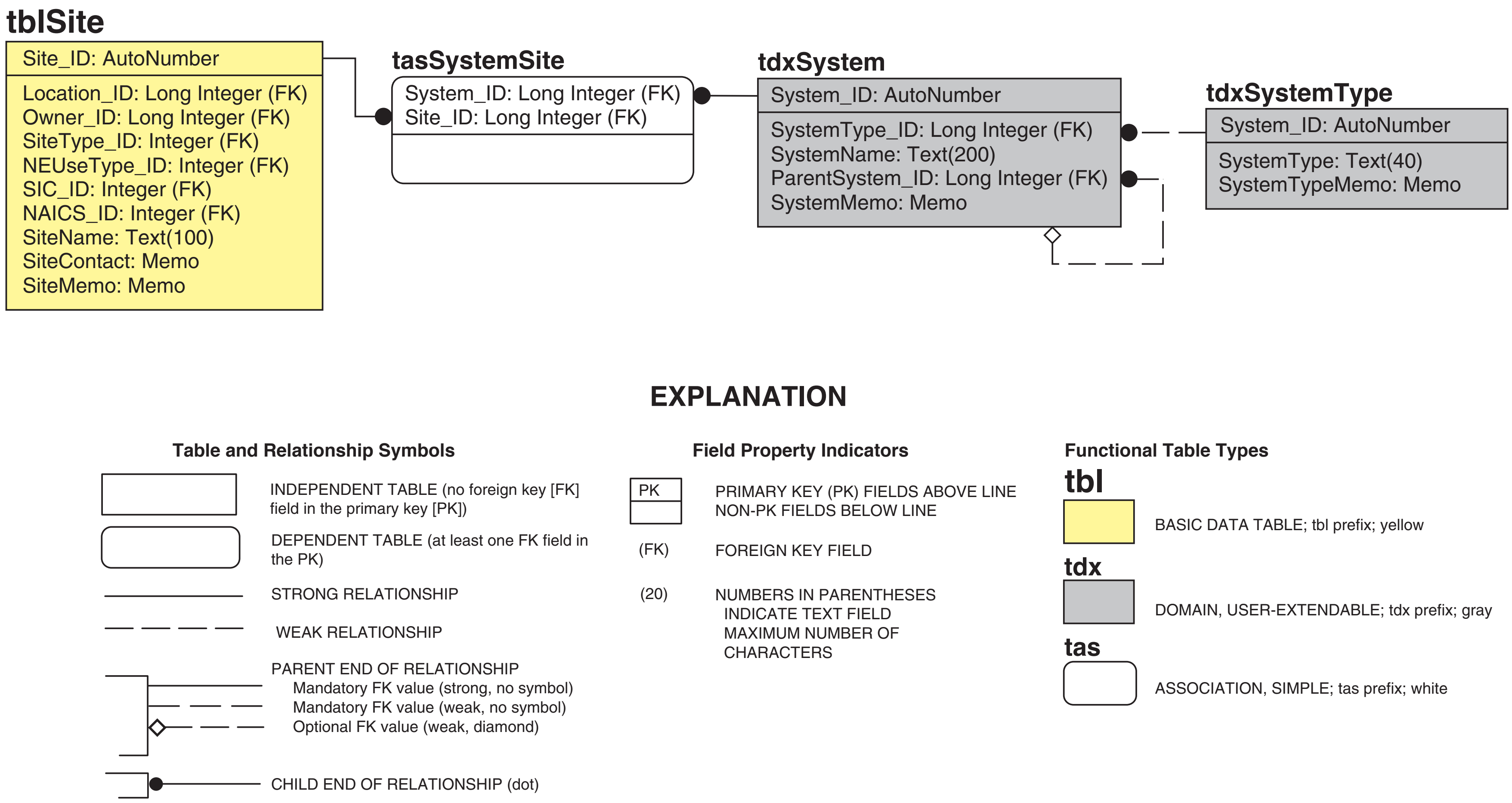
Three tables (*tdxSystem*, *tdxSystemType,* and *tasSystemSite*) are required to set up System-level identities for Sites in NEWUDS, and these tables and their relationships are illustrated in figure 13. System identity information is stored in the table *tdxSystem*. Each System is given a unique *System_ID* and a *SystemName*. A SystemType classification is assigned from the *tdxSystemType* domain, and can include such values as “Public Supplier” and “Public Wastewater System.” An association table, *tasSystemSite*, is used to pair individual Sites with individual Systems. A System can consist of any number of Sites, and each Site can be a member of any number of Systems. This structure allows the NEWUDS user to select all Sites associated with a System or to identify all Systems associated with a Site, and to generate membership statistics, such as the number and type of Sites that belong to three or more Systems.

Additionally, individual Systems can be identified as part of a larger System (a super System) through the hierarchical association of a *ParentSystem_ID* value in the *tdxSystem* with a different System record (a self-join). Thus, a public-supply network could have a System name and consist of two or more named subSystems. Care should be exercised in developing super Systems because individual Sites may be present in more than one subSystem (shared Sites); duplicates should be resolved when handling Systems made up of other Systems.

Not all aggregates of Sites need to be created as named Systems. Using domains and other fields in Site-related tables, any number of useful sets of Sites meeting specific criteria can be assembled, such as those related to use type, industrial classification, Location (geopolitical and hydrologic), Resource properties, Conveyance actions, and Transaction properties. The use of the System tables is reserved for custom, user-defined aggregates of Sites that are identifiable by a convenient name and that need to be manipulated as a group.

## Address Subject Area

The Address subject area focuses on the *tblAddress* table and its related entities. The Address subject area diagram is shown in figure 14. The *tblAddress* table provides a single place to store Address information for Sites and Owners in NEWUDS. Each Address record has standard United States postal fields, and each is classified by a selection from the *tdsAddressTyp*e domain. The *tdsAddressType* domain currently contains three values to accommodate the types of Addresses anticipated: “Street,” “Mailing,” and “Street and Mailing.” One or more types of Addresses for each Site or Owner can be stored by pairing an Address of a defined type with a Site or Owner using the *tasSiteAddress* and *tasOwnerAddress* associative tables. Storing Address information in this way provides data-storage efficiency for the database, in that only Addresses that are known are stored, different types of Addresses can be associated with single Sites or Owners, individual Addresses can serve more than one Site and Owner as appropriate (sharing an *Address_ID*), and fields for storing this optional information are not needed in the *tblSite* or *tblOwner* tables themselves.



**tblSite**

Site_ID: AutoNumber

Location_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
SiteType_ID: Integer (FK)
NEUseType_ID: Integer (FK)
SIC_ID: Integer (FK)
NAICS_ID: Integer (FK)
SiteName: Text(100)
SiteContact: Memo
SiteMemo: Memo

**tasSystemSite**

System_ID: Long Integer (FK)
Site_ID: Long Integer (FK)

**tdxSystem**

System_ID: AutoNumber

SystemType_ID: Long Integer (FK)
SystemName: Text(200)
ParentSystem_ID: Long Integer (FK)
SystemMemo: Memo

**tdxSystemType**

System_ID: AutoNumber

SystemType: Text(40)
SystemTypeMemo: Memo

**EXPLANATION**

**Table and Relationship Symbols**

INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])

DEPENDENT TABLE (at least one FK field in the PK)

STRONG RELATIONSHIP

WEAK RELATIONSHIP

PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)

CHILD END OF RELATIONSHIP (dot)

**Field Property Indicators**

PK — PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE

(FK) — FOREIGN KEY FIELD

(20) — NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS

**Functional Table Types**

**tbl** — BASIC DATA TABLE; tbl prefix; yellow

**tdx** — DOMAIN, USER-EXTENDABLE; tdx prefix; gray

**tas** — ASSOCIATION, SIMPLE; tas prefix; white

**Figure 13.** System subject area tables, fields, and relationships.

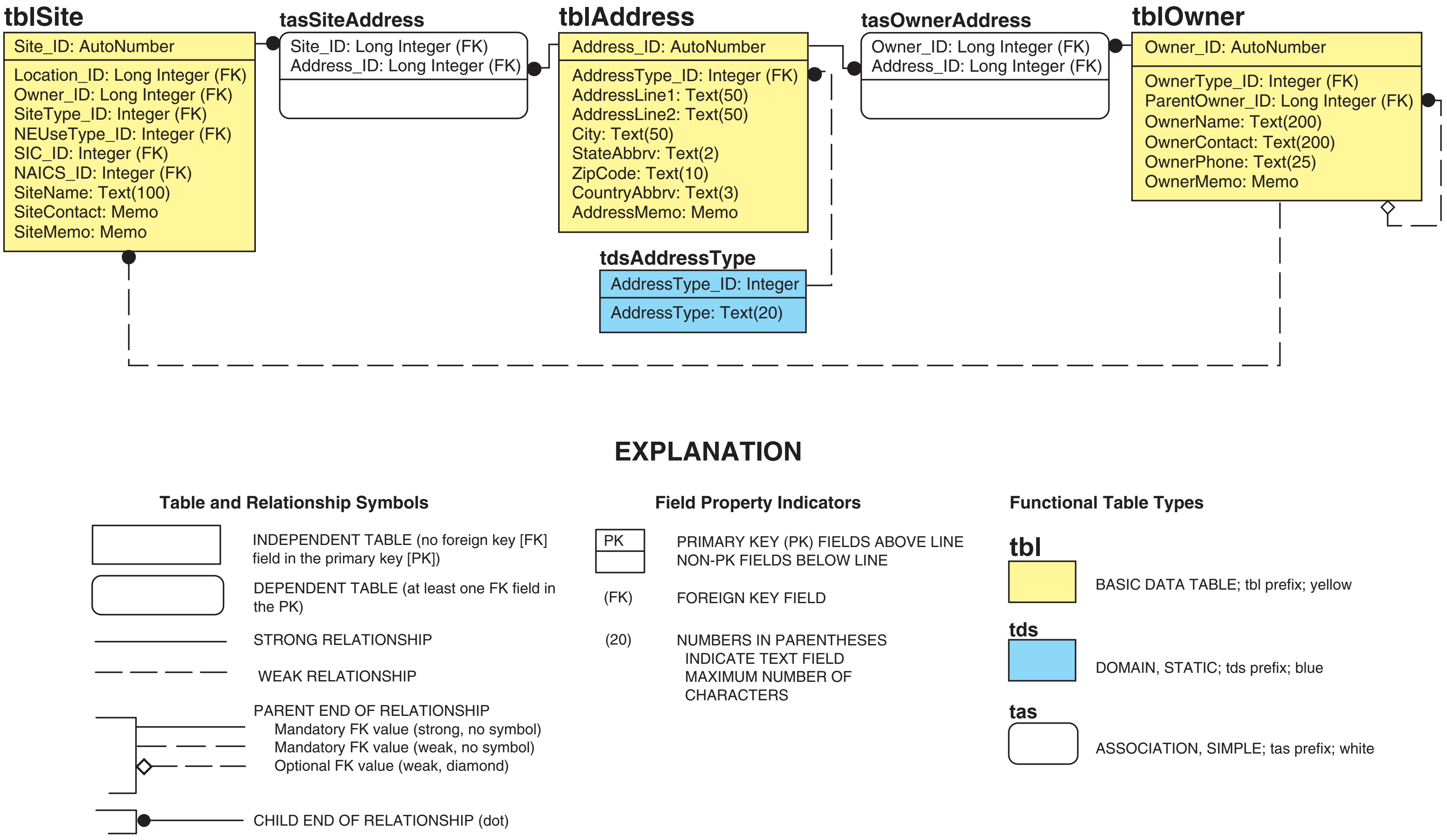

tblSite
Site_ID: AutoNumber
Location_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
SiteType_ID: Integer (FK)
NEUseType_ID: Integer (FK)
SIC_ID: Integer (FK)
NAICS_ID: Integer (FK)
SiteName: Text(100)
SiteContact: Memo
SiteMemo: Memo
tasSiteAddress
Site_ID: Long Integer (FK)
Address_ID: Long Integer (FK)
tblAddress
Address_ID: AutoNumber
AddressType_ID: Integer (FK)
AddressLine1: Text(50)
AddressLine2: Text(50)
City: Text(50)
StateAbbrv: Text(2)
ZipCode: Text(10)
CountryAbbrv: Text(3)
AddressMemo: Memo
tdsAddressType
AddressType_ID: Integer
AddressType: Text(20)
tasOwnerAddress
Owner_ID: Long Integer (FK)
Address_ID: Long Integer (FK)
tblOwner
Owner_ID: AutoNumber
OwnerType_ID: Integer (FK)
ParentOwner_ID: Long Integer (FK)
OwnerName: Text(200)
OwnerContact: Text(200)
OwnerPhone: Text(25)
OwnerMemo: Memo
EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tas
ASSOCIATION, SIMPLE; tas prefix; white

**Figure 14.** Address subject area tables, fields, and relationships.

## DataSource Subject Area

The DataSource subject area focuses on the *tdxDataSource* domain table and its related entities. The DataSource subject area diagram is shown in figure 15. The *tdxDataSource* table is a user-extended domain and provides NEWUDS with a single place to store information about the original source of information for records in various entities. Each DataSource has an Owner (*Owner_ID* is a FK) and a *DataSource* text descriptor field. Because a DataSource is owned by an Owner, some records in the *tblOwner* table are entered only to support a DataSource and not for association with Sites or Conveyances. The *DataSourceMemo* field is provided in *tdxDataSource* for optional notes about each DataSource. DataSource is a mandatory foreign key (*DataSource_ID*) that identifies the original source of data stored for Site and Resource User-Defined Details (in the *tadSiteDetail* and *tadResourceDetail* tables, respectively), and for AliasLabels (*tdxAliasLabel*) and Transaction/Rates (*tblRate*). The *tdxDataSource* table can be used to identify and retrieve records that are associated with a particular DataSource or its parent Owner.

## Alias Subject Area

The Alias subject area focuses on the *tblAlias* table and its related entities. The Alias subject area diagram is shown in figure 16, where the generic *tblAlias* table is linked through associative entities to Sites, Resources, and Conveyances. During NEWUDS development, the need to support alternative naming schemes, especially for Sites, was a high priority. Aliases for Sites, for example, would allow Site information to be displayed and reported using alternative State, Federal, or local codes familiar to users. In addition, Aliases can be useful during batch loading of data through a data entry application.

Initially, two methods of implementing aliasing were evaluated: (1) using separate fields within entities bearing the name of the alias scheme (for example, a field named *NPDESPermitNumber* to store U.S. Environmental Protection Agency (USEPA) National Pollutant Discharge Elimination System (NPDES) numbers); and (2) setting up individual Alias tables in a many-to-one relationship with each core table that required alias coding, and with the Alias table containing the individual fields for each coding scheme. The current system, described below, uses a single *tblAlias* table to serve the aliasing function for all entities and has significant advantages over the other approaches. These advantages include having no Null values in any field used for alternative naming schemes, keeping the number of tables and fields involved in providing complete aliasing services to a minimum, and allowing any number of naming schemes for any number of entities to be implemented using the same structure. Because the *tblAlias* table only contains Aliases that are purposely defined, Sites, Resources, and Conveyances that do not have or need an Alias do not have a record in that table.

The Alias structure works by keeping all Alias values (the alternative name or code) in one table, along with a label selection from the *tdxAliasLabel* domain which includes the source of the Alias (whose naming scheme is being used).  Each entity that requires aliasing services is linked to the *tblAlias* table through an association table, where an individual entity instance is paired with an individual Alias. The association table provides a rapid method for checking to see whether an Alias exists for a given item (by searching for the item's *_ID* value), whereas the *tdxAliasLabel* table allows filtering for Aliases from a specific DataSource or assigned a specific AliasLabel.

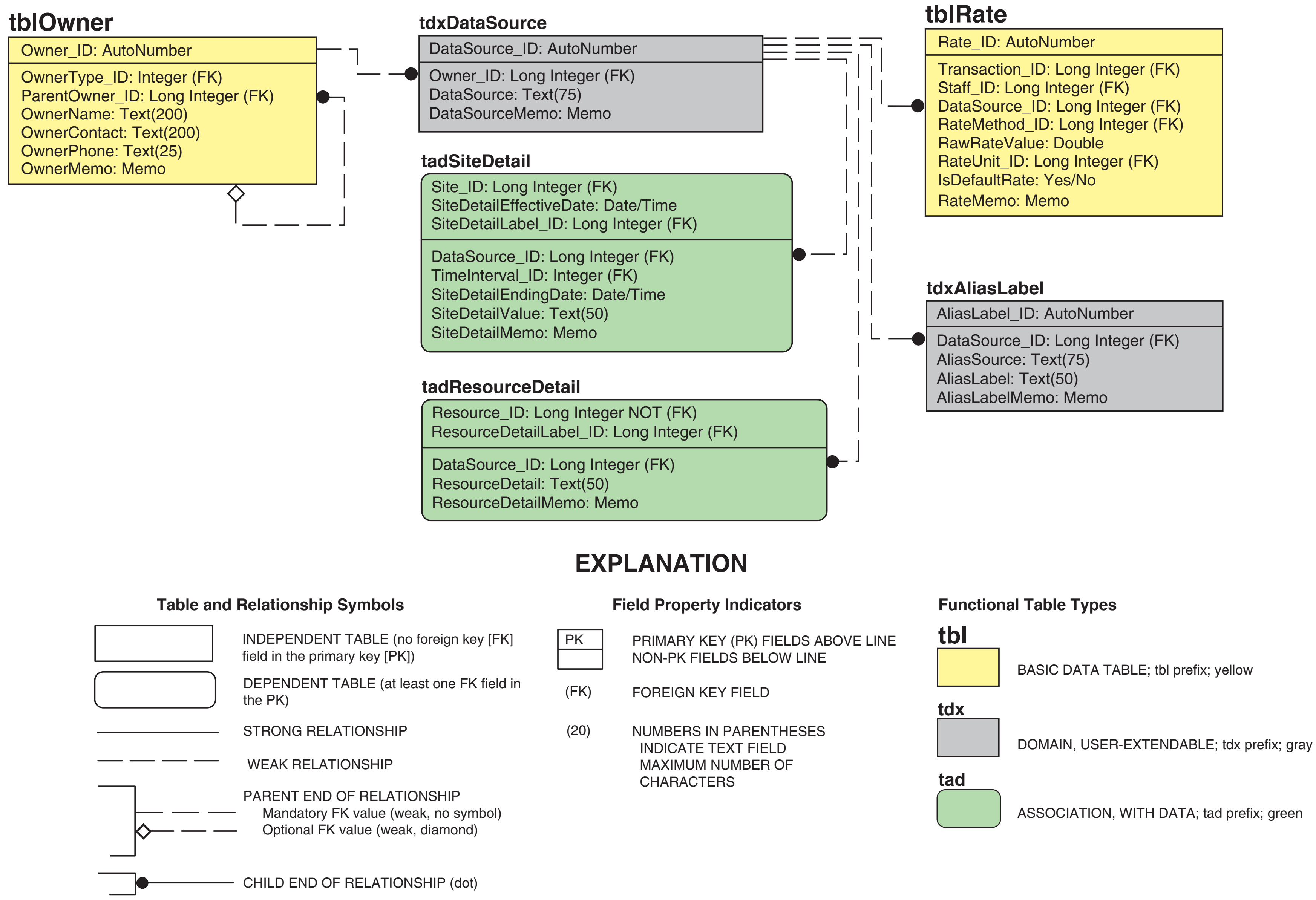

tblOwner
Owner_ID: AutoNumber
OwnerType_ID: Integer (FK)
ParentOwner_ID: Long Integer (FK)
OwnerName: Text(200)
OwnerContact: Text(200)
OwnerPhone: Text(25)
OwnerMemo: Memo
tdxDataSource
DataSource_ID: AutoNumber
Owner_ID: Long Integer (FK)
DataSource: Text(75)
DataSourceMemo: Memo
tblRate
Rate_ID: AutoNumber
Transaction_ID: Long Integer (FK)
Staff_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
RateMethod_ID: Long Integer (FK)
RawRateValue: Double
RateUnit_ID: Long Integer (FK)
IsDefaultRate: Yes/No
RateMemo: Memo
tadSiteDetail
Site_ID: Long Integer (FK)
SiteDetailEffectiveDate: Date/Time
SiteDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
TimeInterval_ID: Integer (FK)
SiteDetailEndingDate: Date/Time
SiteDetailValue: Text(50)
SiteDetailMemo: Memo
tdxAliasLabel
AliasLabel_ID: AutoNumber
DataSource_ID: Long Integer (FK)
AliasSource: Text(75)
AliasLabel: Text(50)
AliasLabelMemo: Memo
tadResourceDetail
Resource_ID: Long Integer NOT (FK)
ResourceDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
ResourceDetail: Text(50)
ResourceDetailMemo: Memo
EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (weak, no symbol)
Optional FK value (weak, diamond)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 15.** DataSource subject area tables, fields, and relationships.

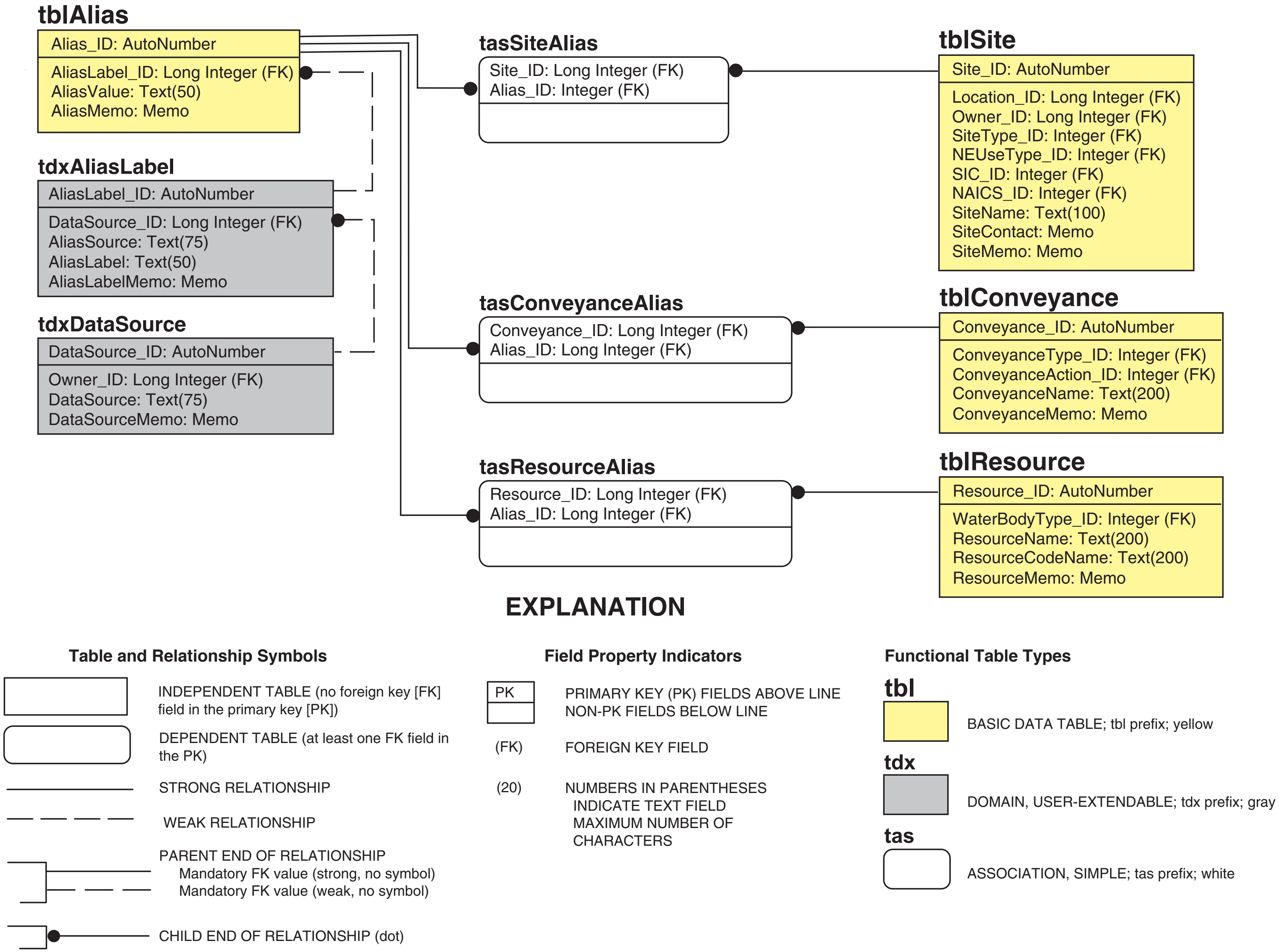

tblAlias
Alias_ID: AutoNumber
AliasLabel_ID: Long Integer (FK)
AliasValue: Text(50)
AliasMemo: Memo
tdxAliasLabel
AliasLabel_ID: AutoNumber
DataSource_ID: Long Integer (FK)
AliasSource: Text(75)
AliasLabel: Text(50)
AliasLabelMemo: Memo
tdxDataSource
DataSource_ID: AutoNumber
Owner_ID: Long Integer (FK)
DataSource: Text(75)
DataSourceMemo: Memo
tasSiteAlias
Site_ID: Long Integer (FK)
Alias_ID: Integer (FK)
tasConveyanceAlias
Conveyance_ID: Long Integer (FK)
Alias_ID: Long Integer (FK)
tasResourceAlias
Resource_ID: Long Integer (FK)
Alias_ID: Long Integer (FK)
tblSite
Site_ID: AutoNumber
Location_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
SiteType_ID: Integer (FK)
NEUseType_ID: Integer (FK)
SIC_ID: Integer (FK)
NAICS_ID: Integer (FK)
SiteName: Text(100)
SiteContact: Memo
SiteMemo: Memo
tblConveyance
Conveyance_ID: AutoNumber
ConveyanceType_ID: Integer (FK)
ConveyanceAction_ID: Integer (FK)
ConveyanceName: Text(200)
ConveyanceMemo: Memo
tblResource
Resource_ID: AutoNumber
WaterBodyType_ID: Integer (FK)
ResourceName: Text(200)
ResourceCodeName: Text(200)
ResourceMemo: Memo
EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; gray
tas
ASSOCIATION, SIMPLE; tas prefix; white

**Figure 16.** Alias subject area tables, fields, and relationships.

In the table *tblAlias*, each Alias is identified by a unique *Alias_ID*. Each Alias is defined by a standardized set of attributes: an *AliasValue*, *AliasLabel*, and *AliasSouce*. An *AliasMemo* field is included for recording optional notes about individual Aliases when necessary. The *AliasLabel* and *AliasSource* are supplied by the *tdxAliasLabel* domain table, which also holds a link to the *tdxDataSource* domain. The *tdxAliasLabel* table also includes a memo field, *AliasLabelMemo*, to record optional descriptions of abbreviated labels or to provide explanations for a label.

Association tables are used to link an Alias to other tables that need Alias assignment. Thus, *tasSiteAlias* links a *Site_ID* to an *Alias_ID*, *tasResourceAlias* links a *Resource_ID* to an *Alias_ID*, and *tasConveyanceAlias* links a *Conveyance_ID* to an *Alias_ID*. Additional associations could be added later for other entities that are not currently provided with aliasing services. Each Alias record will typically serve a single associate, but can apply to more than one and is not limited to any one entity type because it does not store any associate information directly (the linkage is handled in the *tas* association tables). Single items also can have as many Aliases as needed.

An example of using an Alias can be illustrated using the permit numbers for discharge pipes. In NEWUDS, the *Site_ID* for "XYZ discharge pipe" would be connected to the associated Alias (as *Alias_ID*) through the *tasSiteAlias* table. Appropriate field values for defining the Alias could be:

- *DataSource* in *tdxDataSource* = "U.S. Environmental Protection Agency,"
- *AliasSource* in *tdxAliasLabe*l = "PCS" (Permit Compliance System, the USEPA database applicable to the permit number),
- *AliasLabel* in *tdxAliasLabel* = "NPDES permit," and
- *AliasValue* in *tblAlias* = "NH0109999."

Using this example, if all NPDES permit aliases used the same *AliasSource* and *AliasLabel*, it would be possible to construct a query to find all permitted Sites or to locate an individual Site if the permit number was known.

Aliases can be retrieved as a group on the basis of the source or label to supply a different name field to the associated entity. For example, the Alias structure and data can be manipulated through a query to automatically create fields for each label for each aliased entity. An example of using a crosstab query to link Aliases directly to the Site table as extended attributes is given in a later section on View construction.

Using the *AliasSource* and *AliasLabel* fields judiciously, it is possible to store and retrieve an unlimited number of alternative coding systems (by agency, office, or individual) and use them in place of the formal names given in the associated entities. This is much more powerful than setting up separate fields for specific coding schemes. It is expected that over time, however, examination of the *tblAlias* table will indicate which Aliases are truly needed as fields within the *tblSite*, *tblConveyance*, and *tblResource* data tables, and that the *tblAlias* table can then be used to transfer values from those fields to the other tables directly.

One more advantage of using Aliases is that the codes stored could be used to link to other databases. In the discharge permit example above, because the *AliasValue* "NH0109999" would be the same identification used in the USEPA PCS database, a comparison of information, retrievals, and imports from other databases could be facilitated.

## User-Defined Details

The User-Defined Details subject area describes the entity and relationship components that serve the unknown attribute storage needs of several entities. The User-Defined Details subject area diagram is shown in figure 17. As with Alias labels, many specific, optional attributes of various entities in the database were added as fields during the development of NEWUDS and then later removed. A general method was needed to store specific kinds of information that are either optional or not known in advance and could not be designed directly into the table structures. A strategy for storing such data developed over time and resulted in User-Defined Detail structures in NEWUDS.

A User-Defined Detail structure uses a *tad* associative entity to join a specific instance of a core element with a label that describes some detail about it, and stores the detail value using that label. Thus, labels can be created as needed for particular detail value types and reused for other associates. As with Alias labels, careful consideration and occasional reevaluation of the detail labels can provide a mechanism for retrieving sets of similar types of information and for refining searches and aggregations of data in controlled ways. NEWUDS contains User-Defined Detail structures for Sites, Conveyances, Rates, and Resources.

Many details about the various types of Sites in the NEWUDS database can be tracked. Site User-Defined Details may include label values such as “Population served” for public-supply distribution systems or wastewater collection systems; “Status” to store the operational status of a well or treatment plant; “Employees” to store the number of employees at an industrial facility; “Kilowatts generated” to store the power provided by a nuclear facility; or “Cost” to store the amount charged for water by a purveyor.

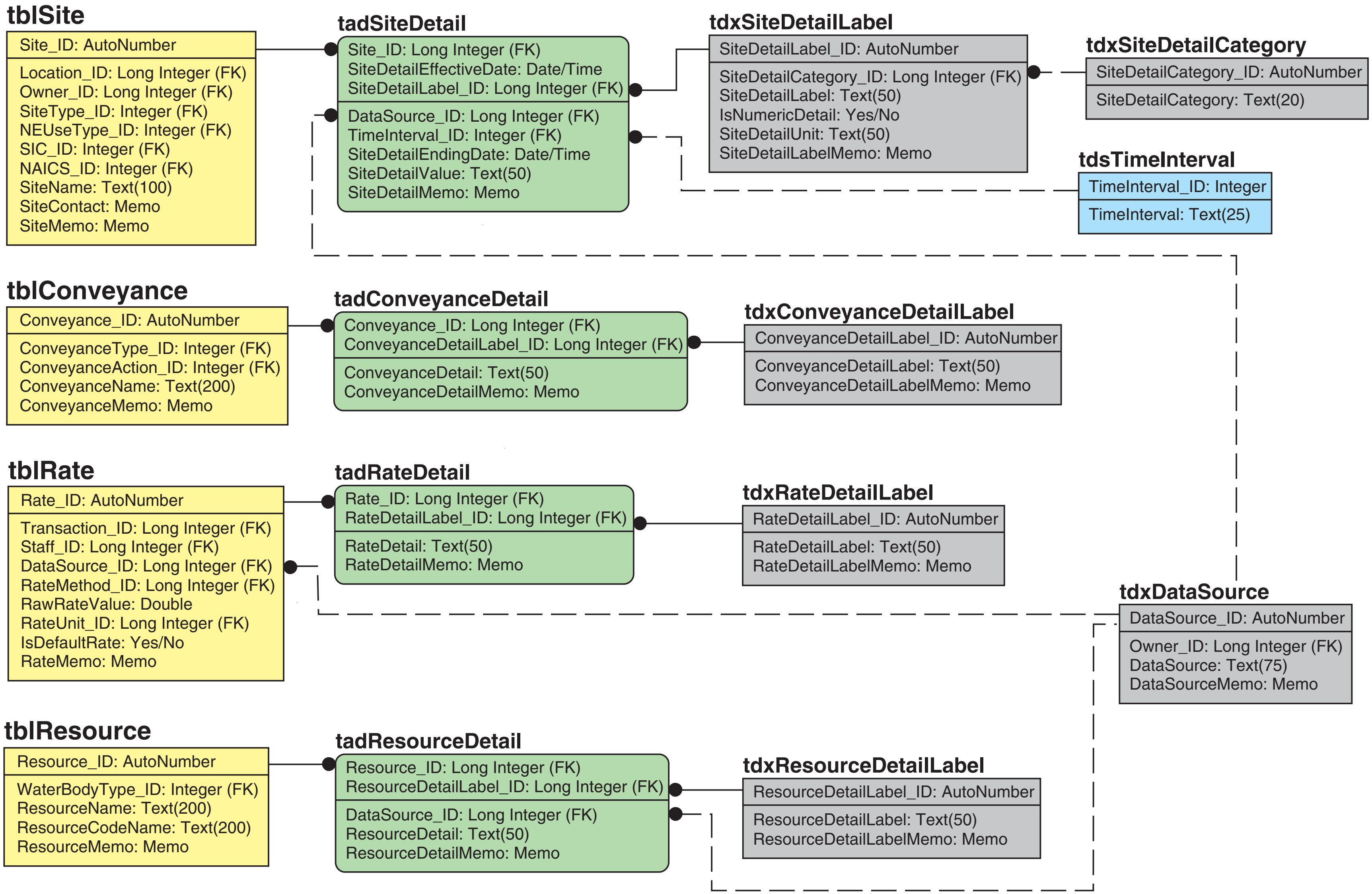
User-Defined Details about Sites may change over time, and temporal fields are needed to define the times covered by the detail values stored. The *tadSiteDetail* table associates a specific *Site_ID* with a specific *SiteDetailLabel_ID* and a specific

*SiteDetailEffectiveDate* to form its primary key. This complex primary key allows the same detail label to apply to a Site over different non-overlapping date periods, thus allowing changes in detail values to be tracked over time (for example, the population served by a wastewater-collection operation). Non-PK fields in *tadSiteDetail* are the *SiteDetailEndingDate* (if known), the *SiteDetailValue* itself, a *SiteDetailMemo* for optional notes about the detail value, and through FKs, the detail *DataSource* and *TimeInterval* classification over which the detail applies.

The *tdxSiteDetailLabel* domain is nested within the *tdxSiteDetailCategory* domain for general detail-label classification information. The *tdxSiteDetailLabel* domain includes a label field, a units field, and a memo field for notes. A field named *IsNumeric-Detail* is used to identify *SiteDetailLabel*s associated with any *SiteDetailValue* that can be converted to a numeric data type for use in mathematical manipulations (for example, the number of employees at industrial Sites).

User-Defined Details for a Conveyance could include construction date, material, capacity, and head-type (pumped or gravity-fed). User-Defined Details about Transaction Rates were mentioned previously, and the structure initially is provided to allow a mechanism to store information on accuracy until a more suitable method is identified. In that case, “Accuracy” would be a *RateDetailLabel* value, and the accuracy to be associated with a particular *RawRateValue* would be entered as the *RateDetail* value. Other labels can be developed as needed.

Resource User-Defined Details have a DataSource associated with them. Examples of details stored for a Resource are the depth or acreage of a reservoir and fisheries classification information.


tblSite
Site_ID: AutoNumber
Location_ID: Long Integer (FK)
Owner_ID: Long Integer (FK)
SiteType_ID: Integer (FK)
NEUseType_ID: Integer (FK)
SIC_ID: Integer (FK)
NAICS_ID: Integer (FK)
SiteName: Text(100)
SiteContact: Memo
SiteMemo: Memo
tadSiteDetail
Site_ID: Long Integer (FK)
SiteDetailEffectiveDate: Date/Time
SiteDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
TimeInterval_ID: Integer (FK)
SiteDetailEndingDate: Date/Time
SiteDetailValue: Text(50)
SiteDetailMemo: Memo
tdxSiteDetailLabel
SiteDetailLabel_ID: AutoNumber
SiteDetailCategory_ID: Long Integer (FK)
SiteDetailLabel: Text(50)
IsNumericDetail: Yes/No
SiteDetailUnit: Text(50)
SiteDetailLabelMemo: Memo
tdxSiteDetailCategory
SiteDetailCategory_ID: AutoNumber
SiteDetailCategory: Text(20)
tdsTimeInterval
TimeInterval_ID: Integer
TimeInterval: Text(25)
tblConveyance
Conveyance_ID: AutoNumber
ConveyanceType_ID: Integer (FK)
ConveyanceAction_ID: Integer (FK)
ConveyanceName: Text(200)
ConveyanceMemo: Memo
tadConveyanceDetail
Conveyance_ID: Long Integer (FK)
ConveyanceDetailLabel_ID: Long Integer (FK)
ConveyanceDetail: Text(50)
ConveyanceDetailMemo: Memo
tdxConveyanceDetailLabel
ConveyanceDetailLabel_ID: AutoNumber
ConveyanceDetailLabel: Text(50)
ConveyanceDetailLabelMemo: Memo
tblRate
Rate_ID: AutoNumber
Transaction_ID: Long Integer (FK)
Staff_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
RateMethod_ID: Long Integer (FK)
RawRateValue: Double
RateUnit_ID: Long Integer (FK)
IsDefaultRate: Yes/No
RateMemo: Memo
tadRateDetail
Rate_ID: Long Integer (FK)
RateDetailLabel_ID: Long Integer (FK)
RateDetail: Text(50)
RateDetailMemo: Memo
tdxRateDetailLabel
RateDetailLabel_ID: AutoNumber
RateDetailLabel: Text(50)
RateDetailLabelMemo: Memo
tdxDataSource
DataSource_ID: AutoNumber
Owner_ID: Long Integer (FK)
DataSource: Text(75)
DataSourceMemo: Memo
tblResource
Resource_ID: AutoNumber
WaterBodyType_ID: Integer (FK)
ResourceName: Text(200)
ResourceCodeName: Text(200)
ResourceMemo: Memo
tadResourceDetail
Resource_ID: Long Integer (FK)
ResourceDetailLabel_ID: Long Integer (FK)
DataSource_ID: Long Integer (FK)
ResourceDetail: Text(50)
ResourceDetailMemo: Memo
tdxResourceDetailLabel
ResourceDetailLabel_ID: AutoNumber
ResourceDetailLabel: Text(50)
ResourceDetailLabelMemo: Memo

Explanation is on next page.

**Figure 17.** User-Defined Detail subject area tables, fields, and relationships.

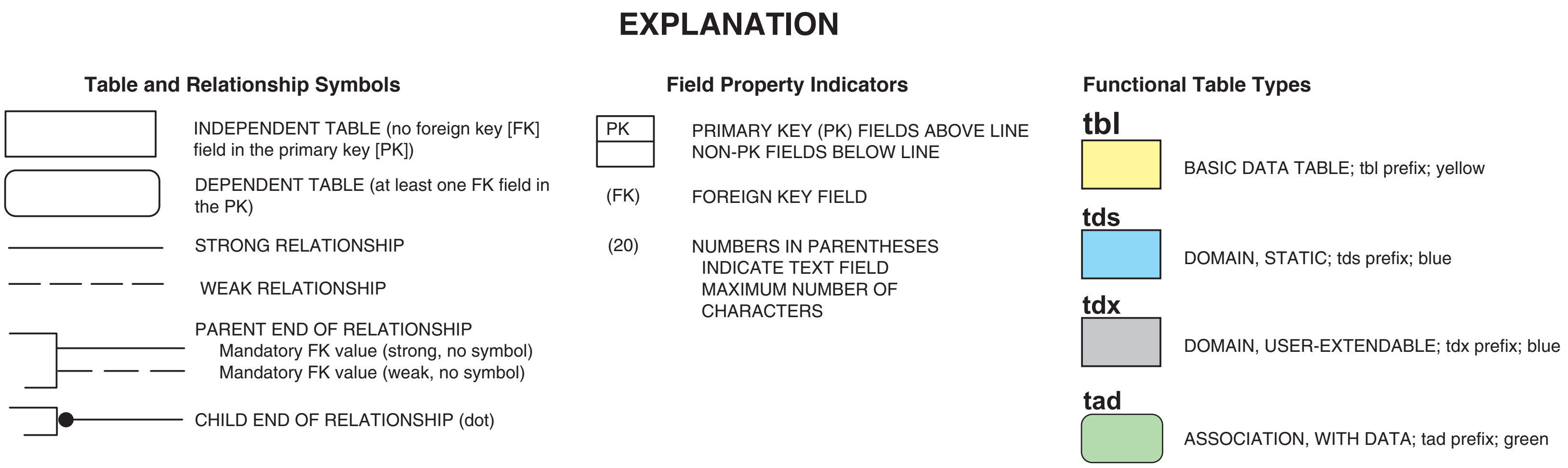

EXPLANATION
Table and Relationship Symbols
INDEPENDENT TABLE (no foreign key [FK] field in the primary key [PK])
DEPENDENT TABLE (at least one FK field in the PK)
STRONG RELATIONSHIP
WEAK RELATIONSHIP
PARENT END OF RELATIONSHIP
Mandatory FK value (strong, no symbol)
Mandatory FK value (weak, no symbol)
CHILD END OF RELATIONSHIP (dot)
Field Property Indicators
PK
PRIMARY KEY (PK) FIELDS ABOVE LINE
NON-PK FIELDS BELOW LINE
(FK)
FOREIGN KEY FIELD
(20)
NUMBERS IN PARENTHESES INDICATE TEXT FIELD MAXIMUM NUMBER OF CHARACTERS
Functional Table Types
tbl
BASIC DATA TABLE; tbl prefix; yellow
tds
DOMAIN, STATIC; tds prefix; blue
tdx
DOMAIN, USER-EXTENDABLE; tdx prefix; blue
tad
ASSOCIATION, WITH DATA; tad prefix; green

**Figure 17.** User-Defined Detail subject area tables, fields, and relationships—*Continued*.

## OPERATIONAL ISSUES AND PROCEDURES

Use of NEWUDS for water-use data requires an understanding of the basic data model, details of the data elements, and the operational aspects of working with data in that specific model design. Operational considerations include (1) creation of indices in an individual implementation of the MS Access database (discussed previously); (2) restrictions on table loading order due to data protection settings, such as forcing selection of classification terms from domain tables; (3) fields that should be automatically updated from other data to facilitate uniformity; and (4) simplification of multi-table structures for handling and presenting data. Several operational strategies have been developed using queries, which are stored in the NEWUDS database as named Structured Query Language (SQL) statements.

### Table Loading Order

Key structures affect table loading order. This information is important for hand-entry of data and for providing structural logic when building a batch loader or user-interface application for NEWUDS. For example, a *tblConveyance* record should not be established without records for the Sites that will form the Conveyance ends already in the *tblSite* table. A *tblConveyance* record can then be added, followed by two entries in *tadSiteConveyance* to create the From and To Site identities for the Conveyance. All static (*tds* prefix) and many user-extendable (*tdx*) domains are already populated in the standard NEWUDS database. Basic data tables (*tbl*) require at least one domain foreign key, so they cannot be loaded until the domain is populated with needed values. Association tables (*tas* and *tad*) require parent table records, and throughout NEWUDS, nesting of table keys establish other loading order requirements.

As a guide for users, loading order information is summarized for each of the 62 NEWUDS tables in table 1. This information should be used to establish proper data entry order and conditions. Columns in table 1 include:

- Loading order—number representing the position of a table in the overall loading order; all tables with a loading order of 1 may be populated immediately; 2’s require at least one table at the 1 level to have been populated with the needed records for the table; 3’s require at least one table at level 2 to have the needed records; and so forth;
- Table name—the name of the table in NEWUDS;
- Inbound foreign keys—the number of foreign key fields coming into the current table from other tables;
- Outbound foreign keys—the number of tables that will receive the primary key of the current table as a foreign key in the child table;
- Pre-loading requirements—the list of tables that need records before the current table (corresponds to the number of Inbound foreign keys) and other prerequisites; and
- Loading notes—things to consider before or during the process of adding records to a table; where “Some fields are optional” is noted, consult the data dictionary for details about optional and required fields in a table.

**Table 1.** Table loading order and related information for New England Water-Use Data System (NEWUDS) tables

[HUC, hydrologic unit codes; MCD, U.S. Census Bureau Minor Civil Division; --, no special considerations]

| Loading order | Table name | Inbound foreign keys | Outbound foreign keys | Preloading requirements | Loading notes |
|---|---|---|---|---|---|
| 1 | tdsAddressType | 0 | 1 | none | Standard NEWUDS contains 3 types |
| 1 | tdsConveyanceActionCategory | 0 | 1 | none | Standard NEWUDS contains 23 categories |
| 1 | tdsConveyanceType | 0 | 1 | none | Standard NEWUDS contains 7 types |
| 1 | tdsHUC | 0 | 1 | none | Standard NEWUDS contains 61 HUCs; must be modified for use outside of New England |
| 1 | tdsLocationScale | 0 | 1 | none | Standard NEWUDS contains 8 scales |
| 1 | tdsNAICS | 0 | 1 | none | Census-based classification should not be modified; contains 1,824 types |
| 1 | tdsOwnerType | 0 | 1 | none | Standard NEWUDS contains 6 types |
| 1 | tdsRateUnitDecimal | 0 | 1 | none | Standard NEWUDS contains 7 decimal variants |
| 1 | tdsRateUnitTime | 0 | 1 | none | Standard NEWUDS contains 6 time variants |
| 1 | tdsRateUnitVolume | 0 | 1 | none | Standard NEWUDS contains 6 volume variants |
| 1 | tdsResourceType | 0 | 1 | none | Standard NEWUDS contains 7 types; could be amended for use outside of New England |
| 1 | tdsSIC | 0 | 1 | none | Census-based classification should not be modified; contains 1,005 types |
| 1 | tdsSiteTypeCategory | 0 | 1 | none | Standard NEWUDS contains 6 categories |
| 1 | tdsState | 0 | 2 | none | Standard NEWUDS contains 7 states; must be amended for use outside of New England |
| 1 | tdsStateBasin | 0 | 1 | none | Not populated for standard NEWUDS; pre-populate as needed with appropriate State basin classification |
| 1 | tdsTimeInterval | 0 | 2 | none | Standard NEWUDS contains 11 interval terms |
| 1 | tdsUSGSUseType | 0 | 1 | none | Standard NEWUDS contains 16 types |
| 1 | tdxConveyanceDetailLabel | 0 | 1 | none | Standard NEWUDS contains 5 labels; populate with appropriate values as needed |
| 1 | tdxLocationDetMethod | 0 | 1 | none | Standard NEWUDS contains 8 methods; populate with appropriate values as needed |
| 1 | tdxRateDetailLabel | 0 | 1 | none | Standard NEWUDS contains 1 label value (“Accuracy”); populate with appropriate values as needed |
| 1 | tdxRateMethodCategory | 0 | 1 | none | Standard NEWUDS contains 11 categories; populate with appropriate values as needed |
| 1 | tdxResourceDetailLabel | 0 | 1 | none | Standard NEWUDS contains 4 labels; populate with appropriate values as needed |

**Table 1.** Table loading order and related information for New England Water-Use Data System (NEWUDS) tables—*Continued*

| Loading order | Table name | Inbound foreign keys | Outbound foreign keys | Preloading requirements | Loading notes |
|---|---|---|---|---|---|
| 1 | tdxSiteDetailCategory | 0 | 1 | none | Standard NEWUDS contains 4 categories; populate with appropriate values as needed |
| 1 | tdxStaff | 0 | 1 | none | Can be pre-populated with intended NEWUDS data entry staff |
| 1 | tdxSystemType | 0 | 1 | none | Standard NEWUDS contains 6 types; populate with appropriate values as needed |
| 2 | tblOwner | 1 | 4 | tdsOwnerType | Some fields are optional |
| 2 | tdsConveyanceAction | 1 | 1 | tdsConveyanceActionCategory | Standard NEWUDS contains 98 actions |
| 2 | tdsCounty | 1 | 2 | tdsState | Standard NEWUDS contains 130 counties; must be amended for use outside of New England |
| 2 | tdsNEUseType | 1 | 1 | tdsUSGSUseType | Standard NEWUDS contains 82 types; tdsNEUseType could be dropped for use outside of New England (regionalized coding) |
| 2 | tdsSiteTypeSubcategory | 1 | 1 | tdsSiteTypeCategory | Standard NEWUDS contains 11 subcategories |
| 2 | tdsWaterBodyType | 1 | 1 | tdsResourceType | Standard NEWUDS contains 10 types |
| 2 | tdxRateMethod | 1 | 1 | tdxRateMethodCategory | Standard NEWUDS contains 36 methods; populate with appropriate values as needed |
| 2 | tdxRateUnit | 3 | 1 | tdsRateUnitDecimal, tdsRateUnitVolume, tdsRateUnitTime | Standard NEWUDS contains 20 RateUnits; a maintenance update query is needed after new records are added |
| 2 | tdxSiteDetailLabel | 1 | 1 | tdxSiteDetailCategory | Standard NEWUDS contains 17 labels; populate with appropriate values as needed |
| 3 | tdsSiteType | 1 | 1 | tdsSiteTypeSubcategory | Standard NEWUDS contains 26 types |
| 3 | tblResource | 1 | 3 | tdsWaterBodyType | Some fields are optional; care should be used to create a ResourceCodeName unique from other resources with the same ResourceName (like “Mill Brook”) |
| 3 | tdsMCD | 1 | 1 | tdsCounty | Standard NEWUDS contains 2,617 MCDs; must be amended for use outside of New England |
| 3 | tdxDataSource | 1 | 4 | tblOwner | Populate with appropriate values as needed; tdxDataSource records are required for adding records to tblRate, tdxAliasLabel, and the detail tables for Sites and Resources |
| 4 | tadResourceDetail | 3 | 0 | tblResource record, tdxResourceDetailLabel, tdxDataSource | ResourceDetail value must be entered |
| 4 | tasOwnerAddress | 2 | 0 | tblOwner and tblAddress records | -- |

**Table 1.** Table loading order and related information for New England Water-Use Data System (NEWUDS) tables—*Continued*

| Loading order | Table name | Inbound foreign keys | Outbound foreign keys | Preloading requirements | Loading notes |
|---|---|---|---|---|---|
| 4 | tasResourceAlias | 2 | 0 | tblResource and tblAlias records | -- |
| 4 | tblLocation | 5 | 3 | tdsLocationScale, tdxLocation DetMethod, tdsState, tdsCounty, tdsMCD | Some fields are optional |
| 5 | tadLocationHUC | 2 | 0 | tblLocation, tdsHUC | IsPrimaryHUC must be entered |
| 5 | tadLocationStateBasin | 2 | 0 | tblLocation, tdsStateBasin | IsPrimaryStateBasin must be entered |
| 6 | tblSite | 6 | 5 | tblLocation, tblOwner, tdsSiteType, tdsNEUse Type, tdsSIC, tdsNAICS | Some fields are optional |
| 3 or 7 | tblAddress | 1 | 2 | tdsAddressType, plus Owner or Site record | Some fields are optional |
| 7 | tadSiteDetail | 4 | 0 | tblSite record, tdsTime Interval, tdxDataSource, tdxSiteDetailLabel | Must enter effective (start) date for detail, and SiteDetail value itself |
| 7 | tadSiteResource | 2 | 0 | tblSite, tblResource | ConnectionCount must be entered |
| 7 | tblConveyance | 2 | 3 | tblSite table for conveyance ends, tdsConveyanceAction | A Conveyance should not be created without having the Sites it connects in tblSite |
| 7 | tdxSystem | 1 | 1 | tdxSystemType and tblSite records must exist | Nodes could be identified and automatically entered using Site and Conveyance information |
| 4, 7 or 8 | tdxAliasLabel | 1 | 1 | tdxDataSource, knowledge of Aliases to be stored | Populate with appropriate values as needed |
| 4, 7 or 8 | tblAlias | 1 | 3 | tdxAliasLabel plus Resource or Conveyance or Site record | Can be created simultaneously with AliasLabel |
| 8 | tadConveyanceDetail | 2 | 0 | tblConveyance record, tdxConveyanceDetailLabel | ConveyanceDetail value must be entered |
| 8 | tadSiteConveyance | 2 | 0 | tblSite and tblConveyance records | FromOrTo field must contain the text “From” or “To” and one of each (two records) is mandatory for each Conveyance |
| 8 | tadSystemSite | 2 | 0 | tdxSystem and tblSite records | Associates could be automatically entered using Site and Conveyance information |
| 8 | tasConveyanceAlias | 2 | 0 | tblConveyance and tblAlias records | -- |
| 8 | tasConveyanceOwner | 2 | 0 | tblConveyance and tblOwner records | -- |
| 8 | tasSiteAddress | 2 | 0 | tblSite and tblAddress records | -- |
| 8 | tblRate | 5 | 1 | tblTransaction, tdxDataSource, tdxStaff, tdxRateUnit, tdxRateMethod | Rate information can be entered simultaneously with Transaction |
| 8 | tblTransaction | 2 | 1 | tblConveyance, tdsTime Interval | Common unit version of default Rate requires a tblRate record and running a maintenance update query |
| 9 | tadRateDetail | 2 | 0 | tblRate record, tdxRate DetailLabel | RateDetail must be entered |
| 9 | tasSiteAlias | 2 | 0 | tblSite and tblAlias records | -- |

## Maintenance Update Queries

Several queries are provided in the NEWUDS database to perform data maintenance functions and to facilitate manipulating data after it has been entered. The queries to perform these functions are run manually or can be triggered to run automatically within an application interface for NEWUDS.

### Rate Unit Updates

The cluster of domain tables centered on *tdxRateUnit* (fig. 9) provide the ability to create RateUnit choices for any possible combination of decimal, volume and time components, as discussed in the Transaction/Rate subject area section of this document. A new RateUnit is added to the domain by selecting one choice from each of the three tables—*tdsRateUnitDecimal*, *tdsRateUnitVolume,* and *tdsRateUnitTime*; an appropriate standard RateUnit abbreviation is entered manually in the field *RateUnitAbbrv*. Two remaining fields are provided with default values of “-1” when new RateUnit entries are added—*RateUnitPhrase* and *MGDConversion*. Although this process establishes the basics of the new record in the *tdxRateUnit* table, one additional step is needed to complete the unit’s description in that table.

The new record in the *tdxRateUnit* table needs to be updated with a standardized component phrase that describes it in the field *RateUnitPhrase* (for example, “thousand cubic feet per month”), and a conversion factor from the new unit to a MGD equivalent needs to be determined and placed in the field *MGDConversion*. The default value for both *RateUnitPhrase* and *MGDConversion* for newly created RateUnits is “-1.” The “-1” value serves as a flag for records that have not yet been updated. Replacing the “-1” flag with a valid conversion value and phrase is done using two queries, one of which sets things up, and the other performs the update. Only the second query needs to be run.

The first query, **qryRateUnitConversionFactor**, creates the phrase and conversion factor from fields provided for that purpose in the three component tables. The phrase is constructed by combining unit terms used in each table: *DecimalUnit* & *VolumeUnit* & “per” & *TimeUnit*. The conversion factor is calculated as the product of the *ConversionToMillion*, *ConversionToGallon*, and *ConversionToDay* fields of the component tables. The SQL statement for this query is:

```
SELECT [tdxRateUnit].[RateUnit_ID], ([tdsRateUnitDecimal].[DecimalUnit] &
IIf(IsNull([tdsRateUnitDecimal].[DecimalUnit]),'',' ') &
[tdsRateUnitVolume].[VolumeUnit] & ' per ' &
[tdsRateUnitTime].[TimeUnit]) AS [RateUnitPhrase],
([tdsRateUnitVolume].[ConversionToGallon]*
[tdsRateUnitTime].[ConversionToDay]*
[tdsRateUnitDecimal].[ConversionToMillion]) AS [MGDConversion]
FROM tdsRateUnitTime INNER JOIN (tdsRateUnitVolume INNER
JOIN (tdsRateUnitDecimal INNER JOIN tdxRateUnit ON
[tdsRateUnitDecimal].[RateUnitDecimal_ID] =
[tdxRateUnit].[RateUnitDecimal_ID]) ON
[tdsRateUnitVolume].[RateUnitVolume_ID] =
[tdxRateUnit].[RateUnitVolume_ID]) ON
[tdsRateUnitTime].[RateUnitTime_ID] = [tdxRateUnit].[RateUnitTime_ID];
```

Note the IIf statement near the beginning of the query. When the decimal component of the constructed unit is one, no explicit statement of that is desirable in the actual unit phrase itself. In the *tdsRateUnitDecimal* table, the textual *DecimalUnit* value for one is left blank (Null). The IIf constraint in the above query adds a space after the decimal term when the *DecimalUnit* is not Null. Using the Null value instead of the term “one” avoids creating an awkward phrase for the unit; for example, the phrase value “cubic feet per second” is the desired phrase compared with “one cubic feet per second.”

The *tdxRateUnit* table is updated by running the query, **qryRateUnitUpdateNEW**. It identifies records where *MGDConversion* = “-1” and applies the above query only to those records needing the update in the *RateUnitPhrase* and *MGDConversion* fields. The SQL statement for this query is:

```
UPDATE [tdxRateUnit] INNER JOIN [qryRateUnitConversionFactor] ON
[tdxRateUnit].[RateUnit_ID] = [qryRateUnitConversionFactor].[RateUnit_ID]
SET [tdxRateUnit].[MGDConversion] =
[qryRateUnitConversionFactor].[MGDConversion],
[tdxRateUnit].[RateUnitPhrase] = [qryRateUnitConversionFactor].[RateUnitPhrase]
WHERE ((([tdxRateUnit].[MGDConversion])= -1));
```

A variant on the second query is provided in NEWUDS to update all the RateUnit phrases and conversion factors in the *tdxRateUnit* table. This is used whenever conversion factors need to be corrected from changes made to the component tables. The query **qryRateUnitUpdateALL** is identical to the query immediately above, but without the constraint of only updating *tdxRateUnit* records where *MGDConversion* = “-1” (the SQL WHERE clause is not used).

### Rate Conversion to Common Units and Updating Transactions

Original Rate data for a Transaction are entered into the *tblRate* table and consist of a value and the units for that value as originally received or created using a specified method and data source. The units are identified in NEWUDS by a selection from the *tdxRateUnit* table. The default Rate value for each Transaction needs to be converted to common units (million gallons per day) and that value has to be entered into the *tblTransaction* table in the field *RateMGD*. Initially, the *RateMGD* value for newly created Transactions is assigned a default value of “-1” and this is used as a flag for values that have not yet been updated. Replacing the “-1” flag with a valid, converted Rate value is done using two queries, one of which sets things up, and the other performs the update. Only the second query needs to be run.

The first query is **qryTransactionConvertDefaultRates** that identifies the default rate entries in *tblRate* for each record in *tblTransaction* and calculates the equivalent converted Rate in Mgal/d common units based on the original RateUnit. The actual calculation is *RawRateValue* * *MGDConversion*. This query is not run, but only provides the set of records representing converted default rates from *tblRate* to the next query. The SQL statement for this query is:

```
SELECT [tblRate].[Transaction_ID], ([RawRateValue]*[MGDConversion]) AS
[ConvertedRate]
FROM [tdxRateUnit] INNER JOIN [tblRate] ON [tdxRateUnit]. [RateUnit_ID] =
[tblRate].[RateUnit_ID]
WHERE ((([tblRate].[IsDefaultRate])=Yes));
```

The second query, **qryTransactionUpdateNEW**, is run to perform the update. This query links *tblTransaction* records where *RateMGD* = "-1" to the first query's set of converted Rate values, and replaces the default "-1" *RateMGD* values in *tblTransaction*. The SQL statement for this query is:

```
UPDATE [tblTransaction] LEFT JOIN [qryTransactionConvertDefaultRates] ON
[tblTransaction].[Transaction_ID] = [qryTransactionConvertDefaultRates].
[Transaction_ID]
SET [tblTransaction].[RateMGD] = [ConvertedRate]
WHERE ((([tblTransaction].[RateMGD])=-1));
```

A variant on the second query is provided in NEWUDS to update all the *RateMGD* values in the *tblTransaction* table. This is to be used whenever a conversion factor for one or more records in *tdxRateUnit* is changed, or when adjustments might be made to *RawRateValues* in the *tblRate* table. The query **qryTransactionUpdateALL** is identical to the query immediately above, but without the constraint of only updating Transaction records where RateMGD = "-1" (the SQL WHERE clause is not used).

### Conveyance Action Phrase Updates

The domain table *tdsConveyanceAction* provides a list of Site-type functions performed on a Conveyance, and has a phrase field (*ConveyanceActionPhrase*) to describe that action (for example, the phrase "From ground-water withdrawal To potable treatment plant"). The phrase is constructed from the same terms used in the *tdsSiteType* table to describe the Sites at the To and From ends of the Conveyance. Exact correspondence between terms used to describe Sites and those used to describe Site connections (Conveyances) is desirable. Although the *tdsConveyanceAction* domain table is considered complete (static), the ability to update the phrase field will accommodate any potential changes to terms used in the *tdsSiteType* table. In the event that additions are made to the *tdsConveyanceAction* domain, the same update facility will create the correct *ConveyanceActionPhrase* (a default value of "-1" is assigned as a flag for new records).

The *tdsConveyanceAction* table contains two pseudo-key fields, *SiteTypeFromID* and *SiteTypeToID*. These fields contain values that correspond to *SiteType_ID* values in *tdsSiteType*. The purpose of the pseudo-keys is to allow transient linkages to be made between the *tdsSiteType* table and the *tdsConveyanceAction* table to automatically update the phrase field. Updating the field *ConveyanceActionPhrase* requires four queries—three to set things up, and one to perform the update. Only the last query needs to be run.

Because there is a one-to-two relationship between a ConveyanceAction and its two SiteType ends, two queries are needed to separately establish SiteType terms for the From and To ends of the Conveyance. A link is made between the *tdsConveyanceAction* and *tdsSiteType* tables, and type terms are collected separately with the text "From" and "To" added. The names and SQL statements for these two queries are:

**qryConvActionSiteTypeFrom**

```
SELECT [tdsConveyanceAction].[ConveyanceAction_ID], ('From ' & [SiteType])
AS [FromSiteType]
FROM [tdsConveyanceAction] INNER JOIN [tdsSiteType] ON
[tdsConveyanceAction].[SiteTypeFromID] = [tdsSiteType].[SiteType_ID];
```

**qryConvActionSiteTypeTo**

```
SELECT [tdsConveyanceAction].[ConveyanceAction_ID], ('To ' & [SiteType]) AS
[ToSiteType]
FROM [tdsConveyanceAction] INNER JOIN [tdsSiteType] ON
[tdsConveyanceAction].[SiteTypeToID] = [tdsSiteType].[SiteType_ID];
```

Next, a query is used to assemble the two parts into a phrase based on their common *ConveyanceAction_ID* values. The query and SQL statement that accomplish this are:

**qryConvActionPhrase**

```
SELECT [qryConvActionSiteTypeFrom].[ConveyanceAction_ID],
([FromSiteType] & ' ' & [ToSiteType]) AS [ConveyanceActionPhrase]
FROM [qryConvActionSiteTypeFrom] INNER JOIN [qryConvActionSiteTypeTo]
ON [qryConvActionSiteTypeFrom].[ConveyanceAction_ID] =
[qryConvActionSiteTypeTo].[ConveyanceAction_ID];
```

The last query applies the results of the previous query to perform the update of the field *ConveyanceActionPhrase* in table *tdsConveyanceAction*. This is the only query that is actually run; the others provide the partial solutions to the update procedure. The query and SQL statement for this procedure are:

**qryConvActionPhraseUpdate**

```
UPDATE [tdsConveyanceAction] INNER JOIN [qryConvActionPhrase] ON
[tdsConveyanceAction].[ConveyanceAction_ID] =
[qryConvActionPhrase].[ConveyanceAction_ID]
SET [tdsConveyanceAction].[ConveyanceActionPhrase] =
[qryConvActionPhrase].[ConveyanceActionPhrase];
```

## Construction and Use of Views

The number of individual NEWUDS tables and the complexity of relationships between them have the potential to be confusing to users. Although the primary reason for most of the individual tables is to respect the rules of normalization and thus provide significant integrity and list-maintenance advantages to the data, the task of manually collecting the correct sets of tables in a query to perform routine exploratory tasks should not be an impediment to using the database. Most of the sets of related tables in NEWUDS subject areas have been pre-assembled into single objects for further manipulation. These general-purpose queries are called Views and are essentially virtual tables; they do not exist as physical file structures, but rather as named SQL statements stored in the database. When run or used in forms or reports, Views create the appearance of and act like a physical table.

A typical View is a query that begins with a base table, then gathers attributes (fields) from related tables and drops the keys that joined them. Thus, fields from attached domain and data tables are added to those of the View base table. Views help users see the data in a familiar flatfile or spreadsheet format for browsing and checking data without the visual distraction of keys and relationships. Views can be combined with other Views to create elaborate View assemblies. In addition, Views can be used to export a physical flatfile directly to file types that can be imported by other applications.

Views are stored as queries with a **qrv** prefix in NEWUDS. In most cases, a View is a complete assembly of nearest-neighbor-related domain tables centered on a single subject area. After a View is created, it can then be combined with others to form much larger data objects that can be handled and whose data can be evaluated at once. In some Views, there is an abbreviated assembly of items due to one-to-many relationships between tables comprising the View. An example of the latter is ignoring non-default Rate values in assembling a View of Transactions (multiple Rate estimates can be stored for a single Transaction, whereas only the default is presented in the View). Likewise, primary HUCs and State Basins are associated with the **qrvLocation** View, rather than creating many new columns for those instances where more than one HUC or StateBasin relates to a single Location. These compromises might affect an analysis and need to be recognized, encouraging custom optimized Views for specific needs.

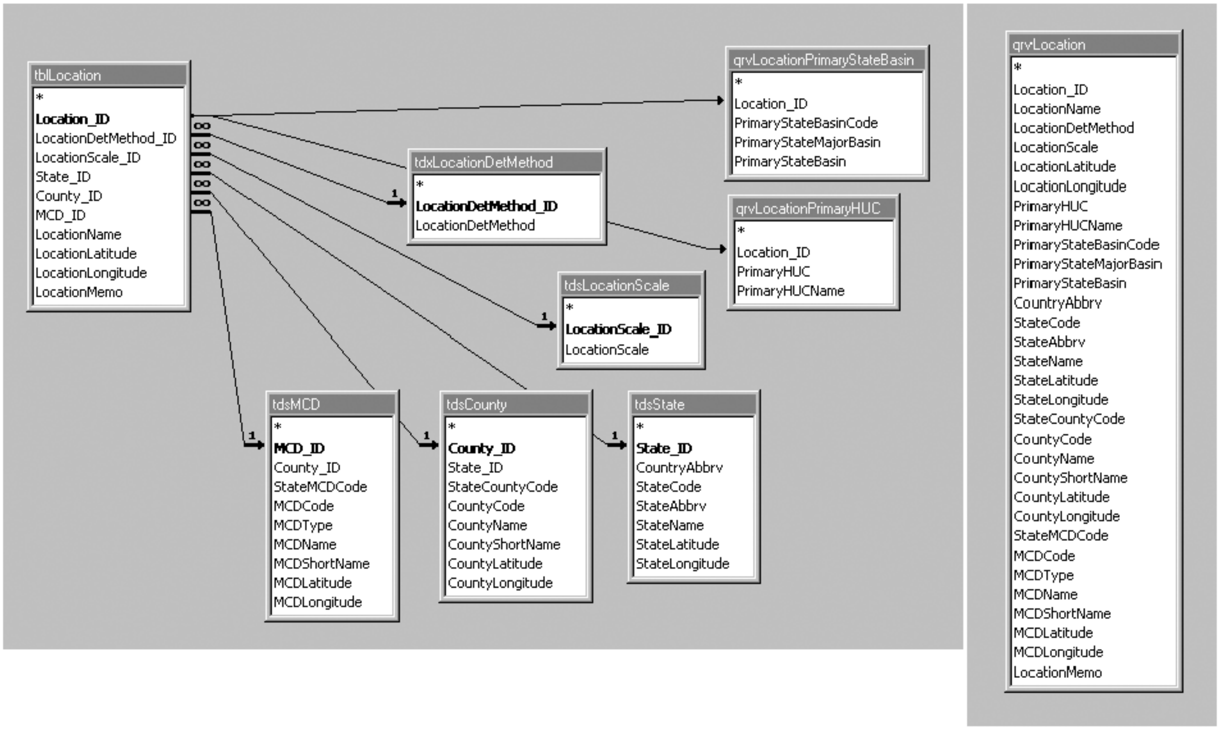
### Example of View Construction

An example of View construction is provided to illustrate the method and result common to all standard NEWUDS Views. The Location subject area in NEWUDS consists of 10 tables (fig. 10). Two of those are associative tables for storing multiple HUCs and StateBasins for a Location, and for each Location only one is designated as the primary. Three queries are needed to create a View of the Location subject area—two to identify the primary HUC and StateBasin (if present) for each Location (**qrvLocationPrimaryHUC** and **qrvLocationPrimaryStateBasin**), and one to incorporate those queries and gather other fields from domain tables with the base *tblLocation* table fields (**qrvLocation**).

The tables and two subqueries involved in the Location View are shown in figure 18 as they would be seen in the upper panel of the MS Access query designer. Note the arrows on the relationship lines that link fields in the base table to the same field in the two subqueries and domains; these indicate that a LEFT JOIN has been established in SQL to ensure that the query returns all Location records, even when no primary HUC or StateBasin entries are present in the two subqueries for a particular *Location_ID* value.

The **qrvLocation** View is built to collect all non-key fields from all joined tables and can use the *Location_ID* to serve as the key for the entire assembly. The result is a single virtual table where each record describes a Location using 30 data fields anchored to the *Location_ID* key. The appearance of the **qrvLocation** View when selected for use in other queries also is shown in figure 18. This virtual table can be used in other queries, manipulated in forms and reports, or exported to spreadsheets or text files without further regard for the complex of 10 original tables and 2 preliminary queries that formed it.

Another type of View is created using a crosstab query in MS Access. Crosstab queries create a table View with new columns based on values in a field and are very useful for denormalizing or rotating data stored in or through association tables. For example, Site Aliases are created by pairing a *Site_ID* with an *Alias_ID* in *tasSiteAlias*, and refer to an entry in *tblAlias*. To determine all the Aliases associated with Sites, one can create a



tblLocation
*
**Location_ID**
LocationDetMethod_ID
LocationScale_ID
State_ID
County_ID
MCD_ID
LocationName
LocationLatitude
LocationLongitude
LocationMemo

qrvLocationPrimaryStateBasin
*
Location_ID
PrimaryStateBasinCode
PrimaryStateMajorBasin
PrimaryStateBasin

tdxLocationDetMethod
*
**LocationDetMethod_ID**
LocationDetMethod

qrvLocationPrimaryHUC
*
Location_ID
PrimaryHUC
PrimaryHUCName

tdsLocationScale
*
**LocationScale_ID**
LocationScale

tdsMCD
*
**MCD_ID**
County_ID
StateMCDCode
MCDCode
MCDType
MCDName
MCDShortName
MCDLatitude
MCDLongitude

tdsCounty
*
**County_ID**
State_ID
StateCountyCode
CountyCode
CountyName
CountyShortName
CountyLatitude
CountyLongitude

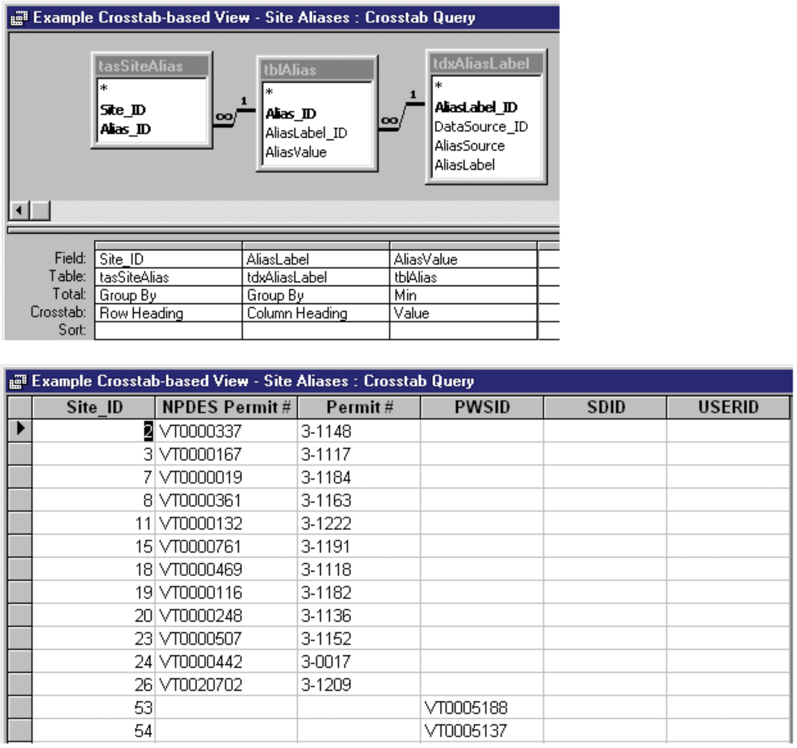
tdsState
*
**State_ID**
CountryAbbrv
StateCode
StateAbbrv
StateName
StateLatitude
StateLongitude

qrvLocation
*
Location_ID
LocationName
LocationDetMethod
LocationScale
LocationLatitude
LocationLongitude
PrimaryHUC
PrimaryHUCName
PrimaryStateBasinCode
PrimaryStateMajorBasin
PrimaryStateBasin
CountryAbbrv
StateCode
StateAbbrv
StateName
StateLatitude
StateLongitude
StateCountyCode
CountyCode
CountyName
CountyShortName
CountyLatitude
CountyLongitude
StateMCDCode
MCDCode
MCDType
MCDName
MCDShortName
MCDLatitude
MCDLongitude
LocationMemo

**Figure 18.** A View assembly (left) and a virtual table (right) focused on the Location table.

crosstab query against the Alias and related tables, and use the *AliasLabel* as the source of the new column labels. The result is a virtual table with *Site_ID* as the key, and a column for each *AliasLabel* associated with Sites. The query layout in MS Access and output for some test data is illustrated in figure 19, where each of the six *AliasLabel*s that were associated with Site Aliases are presented as columns anchored to individual *Site_IDs*. This View can be linked directly to the *tblSite* table through the *Site_ID* to provide the Aliases as extended Site attributes. The column name could be made more informative by concatenating the *AliasLabel* with *AliasSource* or *DataSource* descriptor values.

Views can be linked to other Views to create larger customized flatfile objects for export, use in forms and reports, or for any other purpose. An example of how such a complex object would be assembled is shown in figure 20, where Views of Site, Resource, Owner, and Location are combined to provide a selected subset of 35 Site descriptors from 22 original tables (16 domain, 3 associative, and 3 basic data) anchored to a single *Site_ID* key. The resulting virtual table and a form that uses most of the fields are shown in figure 21. The form demonstrates an exploratory tool for the NEWUDS data model; in MS Access, drill-down filtering on data is possible directly within the form. Custom Views can be constructed and combined with forms to create custom exploratory tools.



Example Crosstab-based View - Site Aliases : Crosstab Query

| | | | |
|---|---|---|---|
| Field: | Site_ID | AliasLabel | AliasValue |
| Table: | tasSiteAlias | tdxAliasLabel | tblAlias |
| Total: | Group By | Group By | Min |
| Crosstab: | Row Heading | Column Heading | Value |
| Sort: | | | |

Example Crosstab-based View - Site Aliases : Crosstab Query

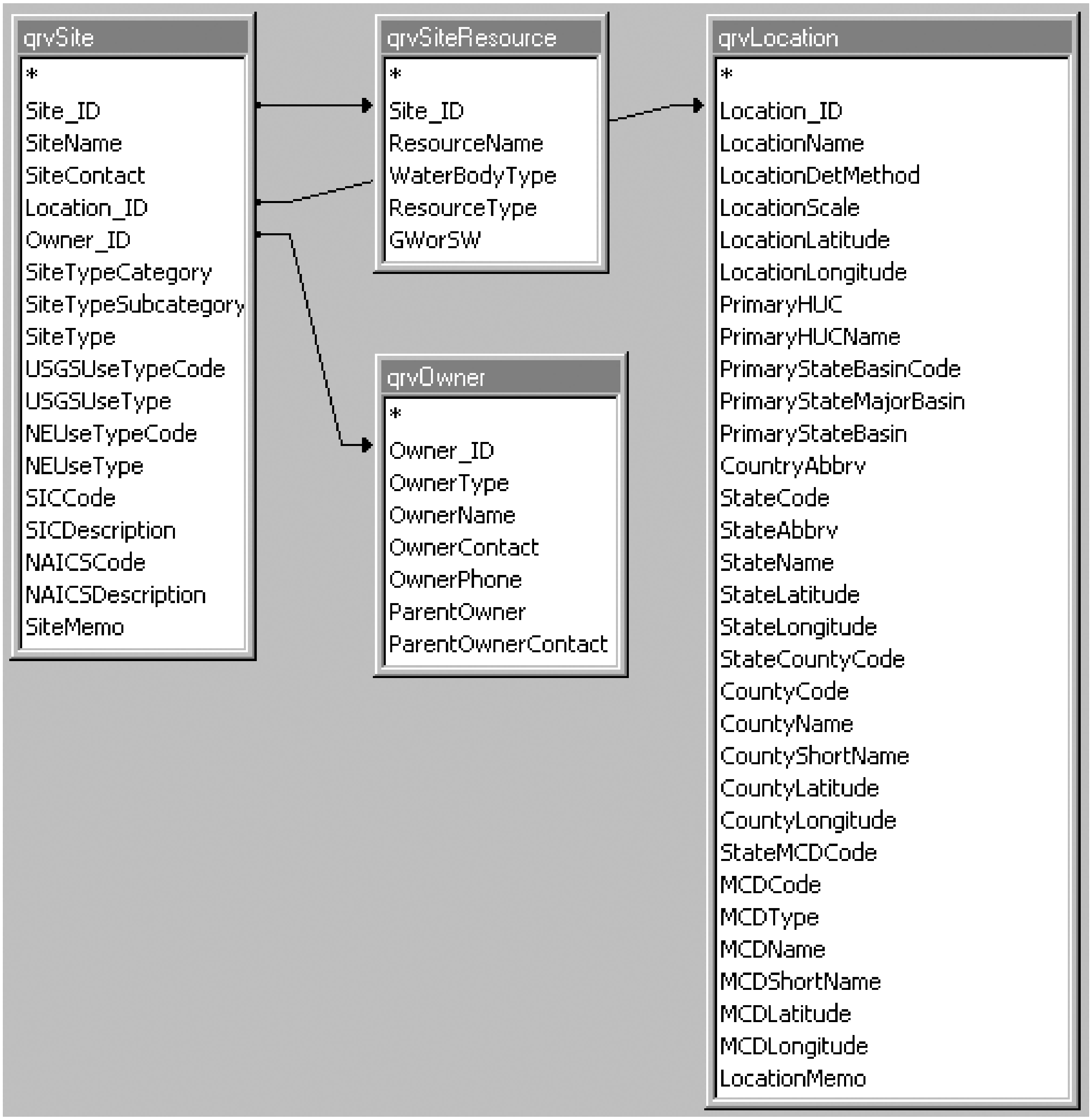
| Site_ID | NPDES Permit # | Permit # | PWSID | SDID | USERID |
|---|---|---|---|---|---|
| 2 | VT0000337 | 3-1148 | | | |
| 3 | VT0000167 | 3-1117 | | | |
| 7 | VT0000019 | 3-1184 | | | |
| 8 | VT0000361 | 3-1163 | | | |
| 11 | VT0000132 | 3-1222 | | | |
| 15 | VT0000761 | 3-1191 | | | |
| 18 | VT0000469 | 3-1118 | | | |
| 19 | VT0000116 | 3-1182 | | | |
| 20 | VT0000248 | 3-1136 | | | |
| 23 | VT0000507 | 3-1152 | | | |
| 24 | VT0000442 | 3-0017 | | | |
| 26 | VT0020702 | 3-1209 | | | |
| 53 | | | VT0005188 | | |
| 54 | | | VT0005137 | | |

**Figure 19.** Illustration showing a crosstab-query design (top) and crosstab-query output (bottom).

### Standardized and Optimized Views

NEWUDS includes 21 preassembled standard Views. These are listed along with their base primary key and a brief description in table 2. These Views are assembled by gathering fields focused on each major subject area in the database. Several Views are based on association tables and can be easily modified for crosstabulation, as described previously, to create denormalized structures for Address, Alias, and User-Defined Detail tables, thereby creating new virtual fields associated with subject area primary keys. The standard Views will produce equivalent output from any independent implementation of NEWUDS (different data, but comparable views).

The standard Views can be used as templates for creating custom optimized queries and views. The standard View contains all value fields from nearby tables and the underlying query might perform sluggishly on large data files. A custom query can drop unnecessary tables, fields, and relationships from a standard View, and this economy will significantly increase performance. Custom queries starting with a View template can rearrange or rename fields and impose conditional criteria, thereby creating precise subsets of data in custom formats. An optimal View would use only those tables and fields that are required for the query purpose—the tables containing the query output fields, and any tables needed to create a relational pathway between those tables.



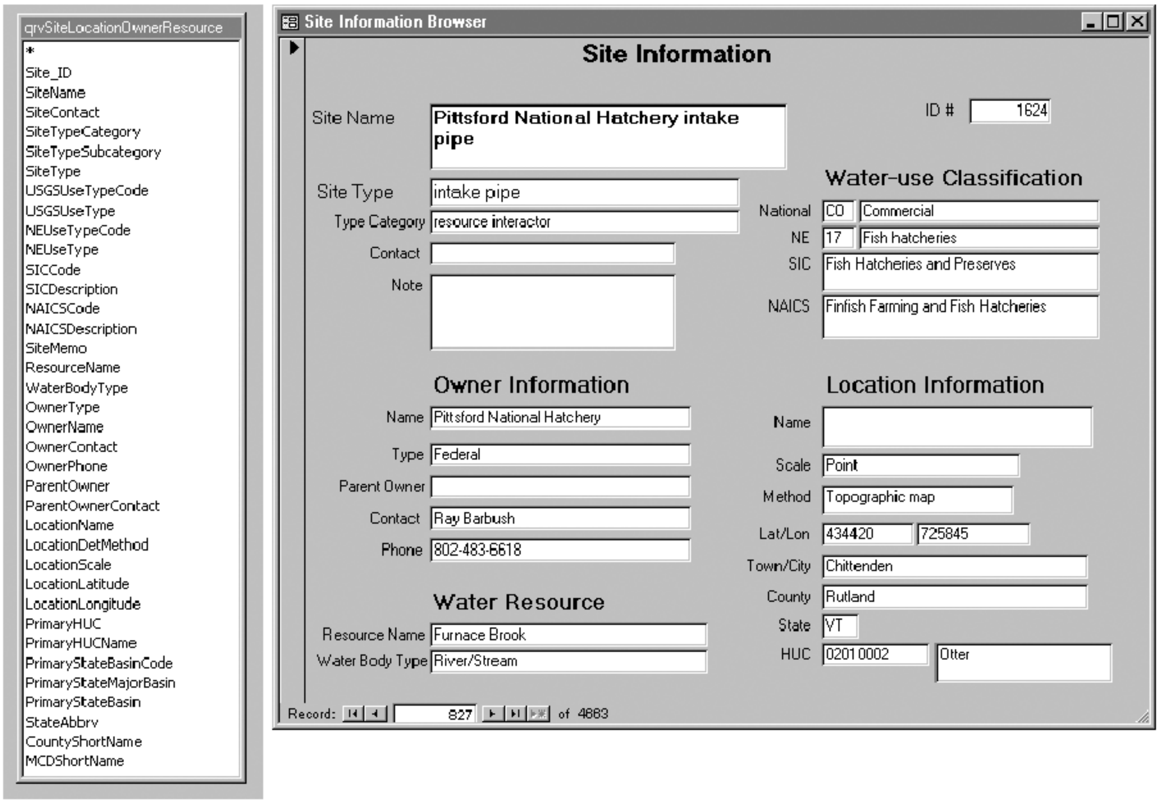
qrvSite
*
Site_ID
SiteName
SiteContact
Location_ID
Owner_ID
SiteTypeCategory
SiteTypeSubcategory
SiteType
USGSUseTypeCode
USGSUseType
NEUseTypeCode
NEUseType
SICCode
SICDescription
NAICSCode
NAICSDescription
SiteMemo

qrvSiteResource
*
Site_ID
ResourceName
WaterBodyType
ResourceType
GWorSW

qrvOwner
*
Owner_ID
OwnerType
OwnerName
OwnerContact
OwnerPhone
ParentOwner
ParentOwnerContact

qrvLocation
*
Location_ID
LocationName
LocationDetMethod
LocationScale
LocationLatitude
LocationLongitude
PrimaryHUC
PrimaryHUCName
PrimaryStateBasinCode
PrimaryStateMajorBasin
PrimaryStateBasin
CountryAbbrv
StateCode
StateAbbrv
StateName
StateLatitude
StateLongitude
StateCountyCode
CountyCode
CountyName
CountyShortName
CountyLatitude
CountyLongitude
StateMCDCode
MCDCode
MCDType
MCDName
MCDShortName
MCDLatitude
MCDLongitude
LocationMemo

**Figure 20**. A complex View assembly made up of other Views.



qrvSiteLocationOwnerResource

*
Site_ID
SiteName
SiteContact
SiteTypeCategory
SiteTypeSubcategory
SiteType
USGSUseTypeCode
USGSUseType
NEUseTypeCode
NEUseType
SICCode
SICDescription
NAICSCode
NAICSDescription
SiteMemo
ResourceName
WaterBodyType
OwnerType
OwnerName
OwnerContact
OwnerPhone
ParentOwner
ParentOwnerContact
LocationName
LocationDetMethod
LocationScale
LocationLatitude
LocationLongitude
PrimaryHUC
PrimaryHUCName
PrimaryStateBasinCode
PrimaryStateMajorBasin
PrimaryStateBasin
StateAbbrv
CountyShortName
MCDShortName

Site Information Browser

Site Information

Site Name: Pittsford National Hatchery intake pipe
ID #: 1624
Site Type: intake pipe
Type Category: resource interactor
Contact:
Note:

Water-use Classification

| | | |
|---|---|---|
| National | CO | Commercial |
| NE | 17 | Fish hatcheries |
| SIC | Fish Hatcheries and Preserves | |
| NAICS | Finfish Farming and Fish Hatcheries | |

Owner Information

Name: Pittsford National Hatchery
Type: Federal
Parent Owner:
Contact: Ray Barbush
Phone: 802-483-6618

Location Information

Name:
Scale: Point
Method: Topographic map
Lat/Lon: 434420 725845
Town/City: Chittenden
County: Rutland
State: VT
HUC: 02010002 Otter

Water Resource

Resource Name: Furnace Brook
Water Body Type: River/Stream

Record: 827 of 4883

**Figure 21.** A custom virtual table (left) created from multiple views, and a form based on the virtual table (right).

## Customizing and Extending the Data Architecture

NEWUDS was designed to be fully extendable and customizable for the specific needs of USGS offices in New England or for individual projects. Two basic ways to extend the database to contain new data are to add fields or to add tables. In either case, extensions to or customizations of the data model should respect existing conventions that apply to normalization, keys and relationships, domain table usage, and naming rules. It is recommended that the names of all custom accessory tables and fields be prefixed with the letter "z" or other unique indicator to easily distinguish them from the fully defined, standard NEWUDS objects.

A field could be added to any table where its data can be expected to be readily available and regularly entered; however, fields should not be added to a working copy of the database to solve short-term needs, such as those that facilitate translating outside data into the NEWUDS format, or to accommodate an unusual data inquiry. In those cases, accessory tables can be created to contain new data fields in a one-to-one relationship with the parent table. The NEWUDS database can then be used with the added data without changing the basic model structure. Fields could be added to the basic model structure when they have proven value, such as Aliases or User-Defined Details that are used frequently as descriptive attributes, or User-Defined Details that demonstrate capabilities to solve analytical problems. NEWUDS may need periodic adjustments based on evaluation of use patterns.

**Table 2.** New England Water-Use Data System (NEWUDS) standard preassembled Views

[HUC, hydrologic unit code; MCD, U.S. Census Bureau Minor Civil Division]

| View (query) name | View primary key | View description |
|---|---|---|
| qrvConveyance | Conveyance_ID | Joins Conveyance action and type domains to Conveyances |
| qrvConveyanceAlias | Conveyance_ID | Assembles Alias data applying only to Conveyances; can be cross-tabbed |
| qrvConveyanceDetail | Conveyance_ID | Joins detail label domain to Conveyance details |
| qrvConveyanceOwner | Conveyance_ID | Assembles Owner data applying to Conveyances |
| qrvLocation | Location_ID | Joins determination method, scale, state, county, MCD, primary HUC and primary state basin domains to Locations |
| qrvLocationPrimaryHUC | Location_ID | Assembles HUCs designated as the defaults for Locations |
| qrvLocationPrimaryStateBasin | Location_ID | Assembles StateBasins designated as the defaults for Locations |
| qrvOwner | Owner_ID | Joins OwnerType domain and parent-owner information to Owners |
| qrvOwnerAddress | Owner_ID | Assembles Address data applying only to Owners |
| qrvRate | Rate_ID (also has Transaction_ID) | Joins method, staff, unit, and data source domains to Rates. |
| qrvRateDetail | Rate_ID | Joins detail label domain to Rate details |
| qrvResource | Resource_ID | Joins resource and water body type domains to Resources |
| qrvResourceAlias | Resource_ID | Assembles Alias data applying only to Resources; can be cross-tabbed |
| qrvResourceDetail | Resource_ID | Joins detail label and data source domains to Resource details |
| qrvSite | Site_ID | Joins site type and four use-type domains to Sites |
| qrvSiteAddress | Site_ID | Assembles Address data applying only to Sites |
| qrvSiteAlias | Site_ID | Assembles Alias data applying only to Sites; can be cross-tabbed |
| qrvSiteDetail | Site_ID | Joins detail label, time interval and data source domains to Site details |
| qrvSiteResource | Site_ID and Resource_ID | Joins views of Sites and Resources |
| qrvSiteLocationOwnerResource | Site_ID | A comprehensive view of Sites constructed from four separate views |
| qrvTransaction | Transaction_ID | Joins time interval domain and Rate view (default values only) to Transactions |

Adding accessory tables to the structure is probably the easiest way to extend the data model, and can provide a means to experiment with new data elements without affecting the data already stored. One-to-zero-or-one (1:0,1) relationships between existing and new accessory tables are the equivalent of adding fields to a table, but those fields are physically stored in a different place. Accessory tables have the added advantages of allowing the new data to be entered only for those records in the parent table that need them and protecting the current data by not altering existing structures. Specialized accessory tables that provide descriptive fields for only some members of an existing table are usually referred to as subtype tables (Fleming and von Halle, 1989, p. 90). For example, the associative table *tadSiteResource* is a subtype table; it contains *Site_IDs* of only resource interactor Sites to pair with a Resource and related domains. As an additional example, to store information specific to only those Sites that are drilled wells, a subtype table named *zWell* could be created to contain fields for construction date, total depth, depth to open interval, and other details that do not apply to Sites in general. The primary key of the *zWell* table would be *Site_ID* implemented as a foreign key from the *tblSite* table, and the only entries in *zWell* would be for Sites (using their *Site_ID*) that are drilled wells and for which the user wants to store the extra information.

By linking to NEWUDS from a separate MS Access database, accessory tables can reside outside of NEWUDS and still be able to use its data. Linking to NEWUDS would be the preferred method for preparing original import data for inclusion into the NEWUDS tables, where a staging area database is created for the import tables and original data manipulations, and new records are passed to the linked NEWUDS tables only after they have been properly quality-assured, formatted, and provided with necessary key values.  An intermediate database as described can be emptied and reused as needed to process new imports.

It is expected that some domain entities may better serve database users if they are extended into more complex domain structures. Nested domain tables, which provide various levels for grouping of domain information, were added during the development of NEWUDS to address this issue (for example, the nested hierarchy of *tdsSiteTypeCategory*, *tdsSiteTypeSubcategory,* and *tdsSiteType* shown in fig. 7). New domains to serve similar functions could be explored. An indication that domain reevaluation is needed would be when an apparent exception presents difficulty in selecting from a domain for particular records in the child table, or when a domain provides too much detail for grouping and sorting purposes (a higher category domain is needed).

In addition, the data model can be simplified by dropping fields from the standard database because they are not being used. That decision should be based on the value of the data to be stored, not on the difficulty of obtaining or entering it. Data elements that are discovered to be ambiguous will require reevaluation to determine if they should be stored at all and whether they are being modeled and stored in the correct way. Absence of data for a required field can always be accommodated with a value representing an absence condition, and nearly all of the domain tables in NEWUDS are equipped with a record for such use.

## CONCLUSIONS

In many areas of New England, withdrawals of freshwater are approaching the operational capacities of developed water supplies.  Local, State, and Federal agencies need data on all aspects of water use to develop comprehensive water-resource management plans and to make decisions regarding water-supply development and requirements for water-conservation measures.  An effective water-resource management plan is contingent upon the data provided by a comprehensive water-use program, and a water-use database is needed to store and retrieve current, accurate, and complete information on what happens to water from points of withdrawal to points of return flow.

The New England Water-Use Data System (NEWUDS) is a database for the storage and retrieval of water-use data.  NEWUDS can handle data covering many facets of water use, including (1) tracking various types of water-use activities (withdrawals, returns, transfers, distributions, consumptive use, wastewater collection, and treatment); (2) the description, classification, and location of places and organizations involved in water-use activities; (3) details about measured or estimated volumes of water associated with water-use activities; and (4) information about data sources and water resources associated with water use.  NEWUDS can be used for large projects (states and regions) and small, focused projects (such as watershed studies).

The core NEWUDS model pairs Sites to form unidirectional Conveyances, and time-bounded water Transactions are stored for the Conveyances. The Conveyance-based model encourages a water network approach to water-use data storage, investigation, and visualization. Information also is stored about Locations, Owners, water Resources, multi-Site Systems, Addresses, DataSources, Aliases, and User-Defined Details.

Standards for normalization, keys and relationships, indices, and naming were applied to the data model and its physical implementation as a stand-alone MS Access database. The database consists of 62 tables, 256 fields, and 74 defined relationships. Five functional types of tables are recognized in NEWUDS and described. Domain tables constitute more than 60 percent of the tables and fields in NEWUDS. The domains are pre-populated with 6,211 classification and descriptive terms and serve as flexible tools for grouping, sorting, filtering, and summarizing information. Operational considerations include table loading order and the use of maintenance update queries and Views. NEWUDS can be customized or extended by adding new fields and tables, by dropping existing fields, and by linking to other databases. The data stored in NEWUDS can be exported or linked to other applications as needed, such as for supply monitoring, statistical analysis, and visualization using Geographic Information Systems (GIS).

NEWUDS was designed to store and retrieve water-use data for New England. Domains may be amended for other areas and for different major water-use patterns. The data contained in a NEWUDS database can be migrated to other data-management tools. Careful management of the quality of data stored using the NEWUDS structure will facilitate accurate, complex investigative queries to assist with management decisions about water use and water resources, and will help avoid obsolescence of the data.

## REFERENCES

Fleming, C.C., and von Halle, Barbara, 1989, Handbook of relational database design: Reading, Mass., Addison-Wesley Publishing Company, 605 p.

Hernandez, M.J., 1997, Database design for mere mortals—A hands-on guide to relational database design: Reading, Mass., Addison-Wesley Publishing Company, 480 p.

Horn, M.A., 2002, in press, User Manual for the New England Water-Use Data System (NEWUDS): U.S. Geological Survey Open-File Report 01-328.

Horn, M.A. and Craft, P.A., 1991, Plan for developing a water-use data program in Rhode Island: U.S. Geological Survey Water-Resources Investigations Report 90-4207, 26 p.

National Institute of Standards and Technology, 1993, Standard for integration definition for information modeling (IDEF1X): Federal Information Processing Standards Publication 184, 155 p., accessed October 20, 2001, on the World Wide Web at URL http://www.itl.nist.gov/fipspubs/0-toc.htm.

Office of Management and Budget, 1987, Standard industrial classification (SIC) manual: Washington, D.C., Executive Office of the President, Statistical Policy Division, 649 p.

Office of Management and Budget, 1997, North American industry classification system (NAICS) – United States, 1997: Washington, D.C., Executive Office of the President, Statistical Policy Division, 1,350 p.

Roman, Steven, 1999, Access database design and programming, (2d ed.): Sebastopol, Calif., O’Reilly and Associates, 409 p.

Solley, W.B., Pierce, R.R., and Perlman, H.A., 1998, Estimated use of water in the United States in 1995: U.S. Geological Survey Circular 1200, 71 p.

U.S. Geological Survey, National handbook of recommended methods for water data acquisition—Chapter 11, Water use, accessed October 20, 2001, on the World Wide Web at URL http://water.usgs.gov/pubs/chapter11/.