PPDM Association

Architectural Principles for PPDM Lite 1.0

April 2004

Revised February 2005

Developed for the PPDM Association by
Wes Baird (Spatial Project Lead)
Members of the PPDM Lite Project

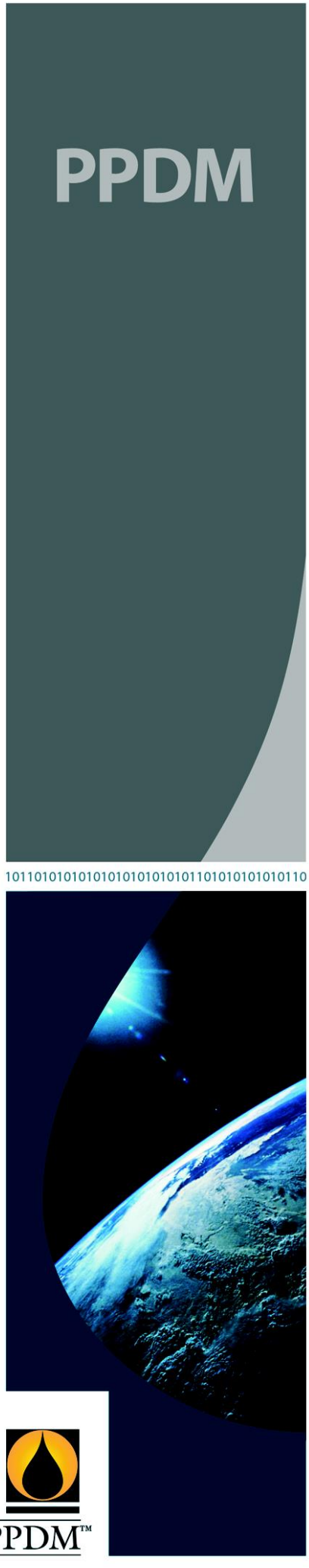

Table of Contents

1. Introduction

The PPDM Lite Architectural Principles document contains the rules and guidelines that govern the development of the PPDM Lite data model. Architectural Principles establish procedures for naming tables, columns and constraints. They also provide guidelines about how columns should be formatted and how common subject areas, such as Geodetics or Units of Measure should be managed.

1.1 Benefits

Architectural Principles are developed to ensure that disparate subject areas within PPDM Lite have a consistent 'look and feel'. These benefits facilitate uptake and implementation of the model for users in resource-based industries, and enable individual users to understand and use the data in the model structure with a high degree of accuracy.

1.2 PPDM Architectural Principles and Open Standards

PPDM work groups or the Modeling Committee make recommendations for additions or enhancement to the Architectural Principles. These are reviewed by the Modeling Committee in light of the impact to the following areas:

- Adherence to open standards
- Impact on users of various data base platforms (Oracle, Sybase, Access etc.)
- Integration or modification required to existing model structures
- Effort needed for development or conversion of software by members
- Effort required for implementation or conversion of data models by members
- Effect on performance, usability and understand-ability of the model
- Cost to implement recommendation in the model

The PPDM Association is committed to the development of an open data model that is not dependent on any vendor or software product for implementation. SQL 92 ANSII standards provide the current foundation for the PPDM Lite version 1.0 Architectural Principles; this enables the Association to utilize open standards that are independent of any Data base vendor.

However, since very few Data base vendors accommodate use of the entire SQL 92 standard, the Architectural Principles are limited to implementation of those standards available to the membership. In practice, the version 1.0 Architectural Principles document supports Entry level SQL 92 compliance.

Additionally, the spatially enabled content of PPDM Lite demands that support be provided for certain proprietary spatial formats, such as Oracle's spatial format, ESRI's SDE binary format and PostGIS's spatial format. This content is included as optional add-ons to PPDM Lite and can be implemented at the user's discretion.

Future design of the data model is likely to require additional data base functionality. Current plans are to review and adopt additional capability as defined in SQL 92 and SQL3 standards as required. Incorporation of significantly changed or enhanced Architectural Principles represents a bench-mark for the PPDM Association in that they represent significant changes that are expected to provide substantial benefits for the membership.

Migration to new models, particularly when new methodologies are employed, can be difficult, time consuming and expensive for the membership. The frequency with which this is done will be minimized by the Association, with careful consideration to the impact among the members. Consequently, significantly changed Architectural Principles will be implemented as part of a new Architectural Release of the model (i.e. version 2.X).

1.3 Integrating PPDM Full and PPDM Lite

Since 1989 PPDM Association has developed a robust, complete data model for the Energy business. The most recent version of this full data model, PPDM 3.7, contains over 1200 tables in 45 subject modules. PPDM Lite is developed primarily to allow implementers to create a standard, light weight, rolled up version of PPDM 3.7 (PPDM full).

PPDM lite contains summary information from a broad spectrum of subject areas, each of which is relatively shallow in it's coverage. Most tables in PPDM Lite contain heavily denormalized summaries that are derived from PPDM 3.7. Mappings between PPDM 3.7 and PPDM Lite are available to members of the PPDM Association.

1.4 Support and Business Modules in PPDM

PPDM consists of an integrated set of business modules that are designed based on the requirements of the work groups and support modules that are designed to enhance and support the information provided by the business modules.

A PPDM work group, through the PPDM modeling process, undertakes Business Module design. Major revisions usually occur when the work group is reconvened to enhance the scope of the model. In contrast, the design of support modules may develop over several modeling cycles through requirements defined by many work groups.

In some cases, a work group may be convened that converts a module from a simple support module into a full-scale business module in its own right.

1.5 PPDM DDL Components

PPDM Data Definition Language (DDL) is provided to members in the following formats:

- Oracle 9.x
- PostgreSQL 7.4

The DDL is separated into a number of files. Each component is described below.

1.5.1 Mandatory Components

Table and column definitions – file extension is .tab

Constraints (primary, foreign, check) – file extension is .con

1.5.2 Optional Components

Table comments – file extension is .tcm

Column comments – file extension is .ccm

Table synonyms – file extension is .syn

Spatial extensions – added using procedures provided by PPDM

2. Change Summary

The following sections of this manual have been added or revised for this version of the Architectural Principles associated with PPDM Lite:

#	Title	Change type
16	*Extensibility*	*Added in the section from 3.7*
	1.X	*Changed 1.x to 1.0 and reviewed for 'Draft' comments*

3. Naming and Design Conventions

3.1 Objectives

Establish naming conventions that provide a consistent reference when accessing components of the model and/or database. Names used should:

- Make sense.
- Be consistent throughout the model.
- Use accepted business terminology where this is very consistent and standard in industry.
- Be named differently than the PPDM 3.7 data model so that PPDM Lite data model can co-exist in the same database schema, if necessary.

Topics include:

Table Design

Column Design

Constraint Design

3.2 Tables

3.2.1 Structure

Naming of tables should reflect the core PPDM 3.7 table name. Ideally, the table name will only be altered to include a 'L_' at the beginning. The structure of a table names is:

L_ +PPDM 3.7 table name

Example:

- **L_WELL (from Well)**
- **L_STRAT_UNIT (from Strat_Unit)**

3.2.2 Structure Rules

1. Maximum of 24 characters is used for PPDM table names.
2. Table names are singular and in present tense.
3. Where the base PPDM 3.7 table is longer than 22 characters, an abbreviation of the last word in the table name should be used to accommodate the 'L_' needed at the beginning of the name.

   Example:

- **SEIS_PROC_STEP_COMPONENT would become L_SEIS_PROC_STEP_COMP**

4. Cross-reference tables created from a single parent table are named by adding an XREF qualifier to the name of the table.

   Example:

   - **L_BA_XREF**

3.2.3 Name Component Rules

1. Names must only contain alphanumeric characters and underscore.
2. Names are not case sensitive.
3. Complete names must not be in the Oracle, Sybase or PL SQL reserved words list (see Appendix A for a list of these words)
4. Avoid using 'A', 'AN', 'AND', 'OF', 'OR', 'THE'.

3.2.4 Consistency

1. Use consistent subject area abbreviations.

Example:

- **SP** – Spatial Project
- **SEIS** - Seismic

2. Use common acronyms or industry accepted abbreviations where useful and meaningful to the membership.

   Example:

   - **AOF** - Absolute Open Flow
   - **GOR** - Gas Oil Ratio
   - **DLS** - Dominion Land Survey

3.2.5 Synonyms

Each table is assigned a name and a synonym in the CASE tool. Synonym names are released as an optional segment of the DDL in a file names *.SYN.

Synonyms are named so as to group tables into logical components. All synonyms will be prefixed with an 'L'.

Primary key and foreign key names are based on the synonyms.

3.2.6 Table comments

1. Every table must have a complete description explaining what it is. If it is a direct copy from PPDM 3.7, then the description will be the same. If it has been altered, then the description will also be altered.

2. Table long names always appear as the first part of the table comments (in upper case), followed by a colon (:).

   Example:

   L WELL SERVICE: The L Well Service table contains common attributes found in all tables that are associated with UWI or well data. This commonality allows a single table to be created in this PPDM Lite data model because the definition of Lite is to provide a quick in-site or overview of data that is available with the understanding that more in-depth data (for analysis) is available in the full PPDM data model.

3.2.7 Table length

Some relational databases impose a limit on the maximum size of a table. SQL Server 2000 limits the total allowed length to 8060 bytes. Even though you can successfully create a table in SQL Server 2000 that could theoretically contain more than 8060 bytes, you will be unable to actually add or update a record with more than 8060 bytes of data. SQL Server 2005 does not have this limitation.

Table design must allow entry level SQL92 compliant relational database implementations to achieve 100% Gold level compliance; for this reason, the total length of a table may not exceed 8060 bytes.

3.3 Columns

PPDM Lite 1.0 column names are based entirely on the source PPDM 3.7 columns that they are derived from. In the case of a module or section being derived from an external data model, best efforts will be made to follow the PPDM Architectural Principles as laid out for PPDM 3.7. This also applies to column comments. In general, the same naming conventions apply for column names as do for table names except that PPDMLite columns are not prefixed by 'L_' and are limited to 27 characters in length.

3.3.1 DATA SOURCE as a column

All PPDM Lite tables contain a reference to the DATA_SOURCE. Source is taken to mean the direct system or application source of the data that was loaded into this row. In the case where PPDM Lite is considered to be a master data store, DATA SOURCE may represent the original source of the data.

4. Constraints in PPDM

Referential Integrity refers to the development of enforced (mandatory or optional) relationships between parent and child tables. In a relational database, the concept of Referential Integrity is critical in order to maintain the value of the information contained within the database. It ensures that database information is associated in a consistent and meaningful fashion.

For example, Referential Integrity prevents a Well Test from being loaded unless the loader can determine which well the test belongs to. It also forces data loaders to validate entries where needed, so that variations or typographic errors in names (Ex. Alberta, Ablerta, Alta, Ab) do not create data retrieval problems later on.

For a detailed discussion of constraints in PPDM, refer to the PPDM Reference Guide on constraints, available on the PPDM web site.

4.1 Primary Key constraints

4.1.1 Objectives

The Primary key of a PPDM table consists of a column or group of columns that uniquely identify each new occurrence or row of data in the table. Cascading Primary Keys In PPDM may also be used to carry key identification values that are commonly queried down into subordinate tables.

Primary keys in PPDM should be enforceable using native SQL only, although triggers or procedures may be necessary to generate surrogate components in some cases.

4.1.2 Guidelines

1. Since the Primary Key components of PPDM tables are mandatory, their data values must be known at load time. Values that may be added later, or not known at load time should not be part of the Primary Key.

2. **Natural identifiers** are used whenever possible as components of the Primary key.

3. **Surrogate Keys** are unique values with little or no intrinsic meaning. In most cases the surrogate key component should be combined with certain natural identifiers to uniquely identify a row. A surrogate key or key component may be used in the following situations.

    - If the number of columns in the natural identifier becomes very large, it may be replaced (all or in part) by a surrogate key - this prevents the migration of an unwieldy key. In this case, the natural identifier must be identified and a unique index created for the natural identifier.

- If the value of the natural identifier is not known at the time the row is created or if there is no natural identifier, surrogate key components such as _ID, OBS_NO or SEQ_NO may be added as a component of the Primary key.

4. All components of the Primary key are mandatory.
5. **DATE** fields should not be used as Primary Key components, since many sites may not know the value of a date at load time.
6. **MEASURED VALUES,** such as DEPTHS, should not be part of the primary key as these create problems when being converted between units of measure.
7. The primary key of a child table may contain components that are inherited from its parent table. Not all components of the parent primary key must cascade into the primary key of the child table. Note that this creates some data management issues; refer to the constraints user guide for details.
8. **Non-static data fields**, which are subject to change or modification, should not be used as Primary Key components. There is considerable overhead associated with managing tables with Primary Keys that may change over time.
9. **Long, multiple component:** Primary keys add some overhead to load and processing time, and can significantly increase the size of subordinate tables, to which the PK is cascaded. Effort should be expended to make the PK contain fewer, rather than more, components.

4.2 Foreign Key constraints

4.2.1 Objectives

Foreign Keys are constructed to ensure that foreign keys can be implemented and enforced using only native SQL whenever possible.

To support good data management practices.

4.2.2 Guidelines

1. In some cases, Designer 2000 projection of tables may result in creation of duplicate columns (i.e. 2 or 3 UWI columns). This is unacceptable, unless it is known that these columns are expected to contain different values (this is unlikely). For these cases, the constraints must be modified to refer to a single column; the redundant columns must be removed from the table definition. This will result in columns that are a component of more than one constraint.
2. Unless the Foreign key columns are part of the Primary key, they should always be defined as NULLABLE.

3. It is acceptable for the PPDM Lite tables to have columns that are bound by more than one constraint. However, it is generally not desirable for a PPDM Lite table to contain multiple columns that refer to the same value. An exception is granted for COMPONENT tables, which may contain foreign key references to many tables, some with the same column name.

4. Value specific exceptions are some of the *_Component tables such as WORK_ORDER_COMPONENT which may have the same value in the COMPONENT_ID and the foreign key such as BUSINESS_ASSOCIATE. The constraint is on BUSINESS_ASSOCIATE, but it is often convenient to also enter the same value into COMPONENT_ID to satisfy the table's Primary Key.

4.3 Check constraints

4.3.1 Objectives

Check constraints can be used to enforce the structural integrity of the data model, or to provide validation for certain kinds of values.

4.3.2 Guidelines

All check constraint values should be provided in UPPER CASE only.

PPDM work groups may, based on the following principles, define constraints:

1. The check constraint may be used to enforce a super-sub type implementation as described in the section on arcs.

2. Values that the column may contain are part of a small set, known at design time, and not subject to change. For example, columns named %_IND may only contain the values Y or N (or NULL).

3. Tables that are projected at the supertype and have a column to specify the subtype that is described. For example, the BUSINESS_ASSOCIATE table uses the column BA_TYPE to define if the row refers to a PERSON, REGULATORY AGENCY, COMPANY or CONSORTIUM.

4.4 Indexes

4.4.1 Unique Indexes

Singleton columns with supporting unique indexes are useful in many circumstances. For example, the spatially enabling methodology created in 2002 – 2003 requires the use of a unique identifier for use in ESRI's Geodatabase.

The PPDM Association has provided early support for globally unique indexes in PPDM version 3.7 through a column called PPDM_GUID (varchar2(38)). The default DDL

scripts will have this column as a null column. There will be an additional script supplied that will:
- convert all PPDM_GUID's to Not Null,
- change the default to an Oracle call that will populate the row upon insertion,
- add a unique index on every column.

DDL commands to do this are included in the PPDM 3.7 data model deliverables. The PPDM compliance measurement process will be modified to ensure that implementations that use this are not penalized for modifying the DDL.

4.4.2 Index On Foreign Keys

PPDM provides a starter set of indexes for every foreign key constraint defined in PPDM. These Indexes should not be considered a final solution, but as a starter to assist in developing a suite of indexes based on site requirements.

We recommend that indexes be placed on the foreign keys, but recognize that too many indexes create their own set of problems. For a complete discussion on indexes in PPDM, refer to the PPDM User Guide on Constraints.

4.4.3 Guidelines

1. PPDM will not attempt to provide a set of performance indexes.
2. PPDM will provide index DDL for foreign key columns. If necessary, PPDM has a script that can be used to generate appropriate indexes and then reverse engineer index DDL from an installed database.

PPDM will not provide index DDL for non-foreign key columns.

4.5 Recursive relationships

Recursive relationships, which allow entries in a PPDM tables to refer to other entries in the same table, occur in two forms:

1. **Recursive relationships (Pig's ears)** - these relationships are stored in the parent table. For example, the WELL table contains a recursive reference to PARENT_WELL
2. **Breakout tables** - these relationships are stored in a separate table, with two relationships / constraints to the same parent table. These relationships may be either 1:1 or 1:m - the implementation is the same.

4.5.1 Guidelines

Methods for querying these structures cannot be developed reliably using SQL 92 standards (see the PPDM User Guide on Constraints for details). In this case, PPDM

allows use of proprietary query mechanisms such as Oracle's CONNECT BY syntax or Sybase's procedural method.

4.6 Naming Conventions

The Association has developed procedures that will modify the names assigned by the CASE tool to conform to the naming conventions in this document.

1. Constraint name may not be the same as a table name.
2. All constraint names must be unique.
3. Constraint name equals the short name (Synonym) of the host table and the short name of the source table concatenated with a key type designation (i.e. **LBA_LWELL_FK** for a foreign key).
4. Foreign key constraint names should indicate the relationship and end in _FK, where n is a sequence number assigned to the foreign constraint with a table in the case where more than one relationship with the parent table exists.
5. Primary key constraint names should be named as SYNONYM _PK. They can be a maximum of 30 characters long.
6. Unique index constraint names should be named as % _UK.
7. Check constraint names should be named as %_CK

5. Design Issues

Please refer to PPDM 3.7 Architectural Principles document for Design Issues.

6. Domains

Please refer to PPDM 3.7 Architectural Principles document for Domain questions.

7. Units of Measure

Please refer to PPDM 3.7 Architectural Principles document for Units of Measure questions.

8. Coordinates

Please refer to PPDM 3.7 Architectural Principles document for Coordinate questions.

9. Extensibility And Sub-setting

Extensibility and subsetting refers to the ability of an individual data model user to customize the model to meet their individual member business needs. In some cases, extensions are needed to manage proprietary or application specific information.

9.1 Objective

To assist in the implementation of extensions or subsets so that changes to the model are easily identified to external users of the model (i.e. application vendors).

9.2 Guidelines

1. Additional tables and columns may be added but the Primary Keys and Foreign Keys of PPDM Lite tables must remain intact.

2. Tables and columns may be subsetted but the Primary Keys and Foreign Keys of remaining PPDM Lite tables must remain intact.

9.2.1 Naming of Extensions

Tables: A total of 6 characters are available for naming private table name extensions. These characters should be added as a prefix to the usual table naming structure.

Example:

- ABC OIL COMPANY wants to extend the model by adding a table containing special core analysis information. They would name the table **AB_**CORE_ANALYSIS where AB_ identifies this a private extension.

Columns: a total of 3 characters are available for naming private column name extensions. These characters should be added as a prefix to the usual column naming structure.

Example:

- ABC OIL COMPANY wants to extend the model by adding a column containing proprietary Seismic line information. They would name the column **AB_**PROPRIETARY_FIELD where AB_ identifies this a private extension.

10. Meta Tables and Meta Data

Please refer to PPDM 3.7 Architectural Principles document for PPDM Meta Tables and Meta Data questions.

11. Target Deliverables

11.1 Documentation

PPDM delivers the following documentation:

- Data Diagrams, showing PK components, FK relationships.
- Table report, a formatted report containing definitions and field descriptions for all tables and columns.
- Mapping between PPDM 3.7 and PPDM Lite 1.0 release.

11.2 DDL

PPDM delivers the following DDL components:

- Table creation.
- Constraint creation (Primary, Foreign, check).
- Units of measure (UOM) and original units of measure (OUOM) foreign key constraints against the meta model.
- Table synonyms for shared use in query development.
- Column and Table comment creation.
- Index creation (note that PPDM provides indexes for all PK and FK components; local DBA tuning will be needed to ensure the correct ones are retained.)