

NOT MEASUREMENT
SENSITIVE

MIL-STD-600006

13 APRIL 1992

MILITARY STANDARD

VECTOR PRODUCT FORMAT



AMSC N/A

AREA MCGT

DISTRIBUTION STATEMENT A. Approved for public release;
distribution is unlimited.

FOREWORD

1. This military standard is approved for use by all Departments and Agencies of the Department of Defense.

2. Beneficial comments (recommendations, additions, deletions) and any pertinent data which may be of use in improving this document should be addressed to: DMA (PRS), 8613 Lee Highway, Fairfax, Virginia 22031-2137, by using the Standardization Document Proposal (DD Form 1426) appearing at the end of this document or by letter.

CONTENTS

<u>PARAGRAPH</u>		<u>PAGE</u>
1.	SCOPE	1
1.1	Scope	1
1.2	Applicability	1
1.3	Classification	1
1.4	Version control	1
2.	APPLICABLE DOCUMENTS	2
2.1	Government documents	2
2.1.1	Specifications, standards, and handbooks	2
2.1.2	Other Government documents, drawings, and publications	2
2.2	Non-Government publications	2
2.3	Order of precedence	3
3.	DEFINITIONS	4
4.	GENERAL REQUIREMENTS	16
4.1	General	16
4.2	VPF characteristics	16
4.3	Relationship between VPF and specific products ..	17
4.4	VPF hierarchy	17
5.	DETAILED REQUIREMENTS	20
5.1	General	20
5.2	VPF data model	20
5.2.1	Data organization	20
5.2.1.1	Directory	20
5.2.1.2	Tables	21
5.2.1.3	VPF table components	23
5.2.1.4	Indexes	24
5.2.1.5	Narrative tables	24
5.2.1.6	Attribute tables	24
5.2.2	VPF data model components	25
5.2.2.1	Primitives	26
5.2.2.1.1	Nodes	26
5.2.2.1.2	Edges	28
5.2.2.1.3	Faces	28
5.2.2.1.4	Text	29
5.2.2.2	Feature classes	29
5.2.2.2.1	Feature definition	29
5.2.2.2.2	Feature table joins	29
5.2.2.2.3	Feature class types	30
5.2.2.2.4	Constructing feature classes	30
5.2.2.3	Coverage	33
5.2.2.3.1	VPF topology	33
5.2.2.3.2	Value description tables	38
5.2.2.3.3	Tiled coverages	38
5.2.2.3.4	Cross-tile keys	39
5.2.2.4	Library	40
5.2.2.4.1	Tile reference coverage	41

CONTENTS

<u>PARAGRAPH</u>		<u>PAGE</u>
5.2.2.4.2	Library attributes	41
5.2.2.4.3	Library coordinate system	41
5.2.2.4.4	Library reference coverage	42
5.2.2.4.5	Data quality reference coverage	42
5.2.2.4.6	Names placement coverage	42
5.2.2.5	Database	42
5.2.3	Data quality	43
5.2.3.1	Types of data quality information	44
5.2.3.2	Data quality encoding	44
5.3	Implementation	44
5.3.1	General implementation information	44
5.3.1.1	Table definitions	44
5.3.1.2	Reserved table names and extensions	46
5.3.2	Primitives	48
5.3.2.1	Node primitives	49
5.3.2.2	Edge primitive	51
5.3.2.3	Face primitive	53
5.3.2.4	Text primitive	55
5.3.2.5	Minimum bounding rectangle table	55
5.3.3	Feature class	57
5.3.3.1	Feature tables	57
5.3.3.2	Feature join tables	58
5.3.3.3	Feature-to-primitive relations on tiled coverages	59
5.3.4	Coverage	59
5.3.4.1	Coverage relationships	59
5.3.4.2	Feature class schema table	59
5.3.4.3	Value description table	61
5.3.5	Library	62
5.3.5.1	Library header table	62
5.3.5.2	Geographic reference table	62
5.3.5.3	Coverage attribute table	62
5.3.5.4	Tile reference coverage	65
5.3.5.5	Tile attributes	65
5.3.6	Database	66
5.3.6.1	Library attribute table	67
5.3.6.2	Database header table	67
5.3.7	Data quality	67
5.3.8	Narrative table	70
5.3.9	Names placement coverage	70
5.4	VPF encapsulation	70
5.4.1	Table definition	70
5.4.1.1	Header	71
5.4.1.2	Record list	73
5.4.1.3	Variable-length index file	73
5.4.2	Spatial index files	74
5.4.3	Thematic index files	76
5.4.4	Allowable field types	78
5.4.5	Naming conventions	78
5.4.6	Triplet id field type	80
5.5	Data syntax requirements	81

CONTENTS

<u>PARAGRAPH</u>		<u>PAGE</u>
5.5.1	Integer numbers	81
5.5.2	Real numbers	82
5.5.3	Date and time syntax	83
5.5.4	Text syntax	84
5.5.5	Coordinate syntax	89
5.5.6	Coordinate strings	89
6.	NOTES	90
6.1	Intended use	90

FIGURES

1	Relationship between VPF and specific products	18
2	Vector product format structure	19
3	Byte stream	20
4	Table types	22
5	VPF structural levels	26
6	Geometric and cartographic primitives	27
7	Primitive directory contents	27
8	Feature class structural scheme	31
9	Coverage contents	33
10	Levels of topology in VPF coverages	34
11	Level 0 topological primitives and their attributes	35
12	Level 1 and Level 2 topological primitives and their attributes	36
13	Level 3 topological primitives and their attributes	37
14	A tiling scheme	39
15	Face cross-tile matching	40
16	Library directory	41
17	Database directory	43
18	Node, edge, and face primitives	48
19	Table structure	71
20	Examples of the triplet id	80
21	Integer number syntax	81
22	Real number syntax	83
23	Date and time syntax	84
24	Latin alphabet primary code table (ASCII)	86
25	Latin alphabet supplementary code table of accents, diacritical marks, and special characters	87
26	Usage of accents and diacritical marks	88

TABLES

1	Directory structure	21
2	VPF table structure	23
3	City attribute table	25
4	State attribute table	25

CONTENTS

<u>TABLES</u>		<u>PAGE</u>
5	Feature table structure	29
6	Feature class schema table	31
7	Feature class schema table and simple feature class	32
8	Columns required to define topology in VPF coverages	35
9	VPF data quality information	43
10	Table definition example	44
11	Field types	45
12	Key types	46
13	Optional/mandatory conditions	46
14	Reserved file names	47
15	Reserved directory names	47
16	Reserved table name extensions	47
17	Entity node definition	49
18	Entity node records	49
19	Connected node definition	50
20	Connected node records	50
21	Edge table definition	52
22	Edge record example	52
23	Face table definition	53
24	Face record example	53
25	Ring table definition	54
26	Ring record example	54
27	Text primitive structure table	55
28	Text primitive record example	56
29	Minimum bounding rectangle definition	56
30	Face bounding rectangle record example	56
31	Feature table definition	57
32	Feature join table definition	58
33	Feature class schema definition	60
34	Feature class schema example	60
35	Value description table definition	61
36	Integer value description record example	61
37	Library header table	63
38	Geographic reference table	64
39	Coverage attribute table definition	65
40	Coverage attribute table example	65
41	Tile reference area feature table definition	66
42	Tile area feature record example	66
43	Library attribute table entity definitions	67
44	Library attribute table example	67
45	Database header table definition	68
46	Data quality table definition	69
47	Narrative table definition	70
48	Header field definitions	72
49	Text header example	73
50	Components of variable-length index file	74
51	Spatial index file header record layout	75
52	Structure of the bin array record	75

CONTENTS

<u>TABLES</u>		<u>PAGE</u>
53	Structure of the bin data record	75
54	Thematic index file header record layout	77
55	Structure of index directory record	78
56	Allowable field types	79
57	Type byte definitions	80
 <u>APPENDIXES</u>		
A	Introduction to the VPF data model	91
B	Winged-edge topology	98
C	Feature class relations	104
D	Tiling	129
E	Data quality	132
F	Spatial indexing	139
G	Coding for metadata tables	151
H	Sample VPF database	154

1. SCOPE

1.1 Scope. The vector product format (VPF) is a standard format, structure, and organization for large geographic databases that are based on a georelational data model and are intended for direct use. VPF is designed to be compatible with a wide variety of applications and products. VPF allows application software to read data directly from computer-readable media without prior conversion to an intermediate form. VPF uses tables and indexes that permit direct access by spatial location and thematic content and is designed to be used with any digital geographic data in vector format that can be represented using nodes, edges, and faces. VPF defines the format of data objects, and the georelational data model provides a data organization within which software can manipulate the VPF data objects. A product specification corresponding to a specific database product determines the precise contents of feature tables and their relationships in the database. In this context, each separate product or application is defined by a product specification and implemented by using VPF structures.

1.2 Applicability. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

1.3 Classification. This standard is UNCLASSIFIED. The procedures and processes presented in this standard may be used for classified processing with the addition of appropriate security provisions.

2. APPLICABLE DOCUMENTS

2.1 Government documents.

2.1.1 Specifications, standards, and handbooks. The following specifications, standards, and handbooks form a part of this document to the extent specified herein. Unless otherwise specified, the issues of these documents are those listed in the issue of the Department of Defense Index of Specifications and Standards (DODISS) and supplement thereto, cited in the solicitation (see 6.2).

SPECIFICATIONS

MILITARY

MIL-D-89009 - Digital Chart of the World Database

HANDBOOKS

MILITARY

MIL-HDBK-850 - DoD Glossary of Mapping, Charting and Geodesy (MC&G) Terms

(Unless otherwise indicated, copies of federal and military specifications, standards, and handbooks are available from the Naval Publications and Forms Center, (ATTN: NPODS), 5801 Tabor Avenue, Philadelphia, PA 19120-5099.)

2.1.2 Other Government documents, drawings, and publications.

This section is not applicable to this standard.

2.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are DOD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation (see 6.2).

INTERNATIONAL STANDARDS ORGANIZATION (ISO)

- ISO 646 - Information Processing-ISO 7-Bit Coded Character Set for Information Exchange.
- ISO 2022 - Information Processing-ISO 7-Bit and 8-Bit Coded Character Sets-Code Extension Techniques.
- ISO 2375 - Data Processing-Procedure for Registration of Escape Sequences.

- ISO 6937 - Information Processing-Coded Character Sets for Text Communication.
- ISO 8601 - Data Elements and Interchange Formats - Information Interchange-Representation of Dates and Times.
- ISO 9660 - Information Processing-Volume and File Structure of CD-ROM for Information Interchange.

AMERICAN NATIONAL STANDARDS INSTITUTE/INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (ANSI/IEEE)

- ANSI/IEEE 754-1986 - IEEE Standard for Binary Floating Point Arithmetic.

AMERICAN NATIONAL STANDARDS INSTITUTE (ANSI)

- ANSI X3.4-1977 - Code for Information Interchange (ASCII) Adopted in FIPSPUB 1-1, 24 December 1980.

Note: VPF products may reside on a variety of media. If a VPF database product resides on CD-ROM, ISO 9660 CD-ROM format is required for that product.

(Copies of ISO and ANSI documents are available from the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

(Non-Government standards and other publications are normally available from the organizations that prepare or distribute the documents. These documents also may be available in or through libraries or other informational services.)

2.3 Order of precedence. In the event of a conflict between the text of this document and the references cited herein, the text of this document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

3. DEFINITIONS

Area feature. A geographic entity that encloses a region; for example, a lake, administrative area, or state.

Area feature class. A collection of area features that maintains a homogeneous set of attributes. Implies the use of face primitives.

Area feature table. The implementation of an area feature class in a VPF attribute table.

Attribute. A property of an entity; for example, the color of a building, the width of a road, or the accuracy level of a database. Defined subtypes of an attribute are the feature attribute, coverage attribute, database attribute, and library attribute.

Attribute accuracy. Attribute accuracy refers to the accuracy or reliability of attribute data within the limits described by feature completeness. If attribute accuracy information is not available in the above form, a description of known attribute accuracy characteristics may be substituted.

Attribute completeness. Attribute completeness refers to the percentage of feature attribute fields not populated by null or default values.

Attribute table. A collection of identically formatted (defined) attribute rows. An attribute table inherits the properties of a VPF table, but also may have value description tables.

Attribute value. The specific value of an attribute; for example, green for building color, 48 feet for road width, level 2 for the accuracy of a database.

Byte order. A hardware implementation of an encoding scheme. It determines the order in which bytes are stored in a long word. Two commonly used orders are little-endian, or least significant first (i.e., 1234); and big-endian, or most significant first (i.e., 4321).

Cartographic primitive. The text primitive.

Code table. A set of character specifications. A code table defines the alphanumeric and special characters that are used in a computer system to model written languages.

Column. The set of all values of a particular attribute within a table.

Column type. The relational model uses column types to implement the data type of an attribute. For instance, the column ELEVATION could have an integer column type.

Complex feature. A feature that includes simple or other complex features. For instance, a watershed complex feature may include river (line), lake (area), and well (point) simple features.

Complex feature class. A feature class that includes two or more other feature classes (simple or complex).

Complex feature table. An implementation of a complex feature class in VPF.

Compound key. A group of columns used together to create a key in a relational table.

Connected node. One of the two node primitive types used to represent linked features that are zero dimensional at a particular scale. Connected nodes are always found at the ends of edges and are topologically linked to the edges. Connected nodes are used in two ways: (1) to define edges topologically (always) and (2) to represent point features that are found at a juncture of linear features, such as overpasses, locks in a canal, or underground utility access points. Under the first usage, the connected nodes are referred to as start and end nodes. Under the second usage, attributes will be associated with the point features related to the connected nodes. If many edges intersect a node, only one edge will be maintained per node in the connected node table; other edges are linked by using winged-edge topology.

Containing face. A face that contains one or more entity nodes. Used to establish the relationship from a node to its face, if level 3 topology is present.

Coordinate. A specified position in Cartesian space. The value takes the form of a short or long floating point value. Z values (if any) are ignored during the enforcement and use of planar graphs.

Coordinate array. A fixed-length list of coordinate tuples.

Coordinate pair. A specified position in a two-dimensional grid, where the first position relates to the X axis and the second position relates to the Y axis.

Coordinate string. A variable-length list of coordinate tuples.

Coordinate triplet. A specified position in a three-dimensional grid, where the first position relates to the X axis,

the second position relates to the Y axis, and the third position relates to the Z axis.

Coordinate tuple. A coordinate pair or triplet.

Coverage. A set of feature classes that has a specified spatial extent and in which the primitives interconnect as described by the coverage's topology.

Coverage attribute. A property of a coverage. The coverage attribute table contains properties for all coverages in the library.

Cross-tile topology. The encoding of topological relationships in such a manner that those relations are maintained even when a coverage has been physically partitioned into multiple tiles.

Data dictionary. A collection of tables with entries that define the meaning of attributes and the allowable values (or ranges of values).

Data syntax. A description of the computer-readable (bit-level) representation of data.

Database. A collection of related libraries as defined by a product specification.

Database attribute. A property of a database.

Date status. Date status refers to the date at which the data was introduced or modified in the database. This date of entry is used as a proof of modification for a single data element and permits the statistical interpretation of groups of data elements.

Direct access. Retrieval of data by reference to its location on a storage medium rather than relative to the previously retrieved data. The access mechanism goes directly to the data in question. This access method is normally required for online data usage.

Directory. A file that contains a list of the unique names, file types (directory or table), beginning addresses, and lengths of other files.

Edge. A one-dimensional primitive used to represent the location of a linear feature and/or the borders of faces. Depending upon the level of topology, edges may be topologically linked to nodes, edges, and faces. Edges are composed of an ordered collection of two or more coordinate tuples (pairs or triplets). At least two of the coordinate tuples must be

distinct. The orientation of an edge can be recognized by the ordering of the coordinate tuples.

Encapsulation. A set format that serves to identify data elements.

Encoding. The assignment of bit-patterns to data types in a computer. For example, one given bit arrangement may define an integer data type (e.g., 2's complement, 1's complement, or biased), whereas another may describe a character data type (e.g., ASCII, EBCDIC).

End node. The terminating node of an edge.

Entity. A general term for any object that is being modeled or defined within a database.

Entity node. One of the two node primitive types used to represent isolated features that are zero dimensional at a particular scale. Entity nodes are topologically linked to a containing face when face topology is present. Entity nodes can reside at any location, whether or not there is another primitive at that same location.

Escape sequence. A special character code used to extend the characters used in a character code table.

Face. A region enclosed by an edge or set of edges. Faces are topologically linked to their surrounding edges as well as to the other faces that surround them. Faces are always nonoverlapping, exhausting the area of a plane.

Feature. A model of a real world geographic entity. A zero-, one-, or two-dimensional entity of uniform attribute scheme from an exhaustive attribute distribution across a plane, or a set of such entities sharing common attribute values. The three subtypes are simple features, complex features, and text features. The types of simple features are point features, line features, and area features.

Feature attribute. A property of a feature.

Feature class. A set of features that shares a homogeneous set of attributes. A feature class consists of a set of tables that includes one or more primitive tables and one or more attribute tables. A feature class has the same columns of attribute information for each feature. Every feature class has one and only one feature table. The three types of feature classes are the simple feature class, complex feature class, and text feature class. The types of simple feature classes are the point feature class, line feature class, and area feature class.

Feature class schema table. A table that stores the composition rules of each feature class. This table describes the definition for each feature class and the way in which each table in a feature class relates to other tables.

Feature completeness. Feature completeness refers to the degree to which all features of a given type for the area of the data set have been included.

Feature join table. A table that identifies 1:N or N:1 relationships between features and other features or primitives. Simple features may be composed of one or more primitive instances, and complex features may be composed of one or more simple features or other complex features. A primitive instance may belong to more than one feature.

Feature table. A table made up of rows of features in a feature class. These rows collectively form the feature table for that feature class.

Field. A field contains a single attribute value of a single entity. Fields in a table identify the data types contained within each table.

File. A named stream of bytes. The three VPF file types are directory, table, and index file.

First edge. An edge arbitrarily selected as the first edge to enable traversing around a connected node, or through the ring table associated with a face.

Fixed field. A field made up of a predefined number of bytes. Fixed fields are generally used for numeric data, or when blank entries are significant.

Foreign key. One or more columns in one table that are used as a primary key in another table.

Names placement coverage. A coverage that contains (at a minimum) a point feature table with columns indicating a place name and its known coordinate. Used to help a user locate places by name.

Geographic entity. A phenomenon characterized by its locational context and about which spatially referenced information is stored.

Geographic information system (GIS). An organized collection of computer hardware, software, geographic data, and standard operating procedures for efficiently capturing, storing, maintaining, retrieving, analyzing, displaying, and reporting spatially referenced information.

Geographic reference table. A table that defines the coordinate system of a library.

Geometric primitive. The basic geometric units of representation; specifically, nodes, edges, and faces.

Georelational data model. A generic conceptual model in which geographic information is represented by using a combination of vector geometry and planar, topology, and relational data models.

Index. A mechanism used to quickly identify a particular record or group of records based on a table's primary key.

Index file. The implementation of an index stored in a file instead of a table. There are row id indexes, variable-length indexes, spatial indexes, and thematic indexes.

Inner ring. The inner boundary of a face, composed of edges ordered in a sequence. A face may have none or any arbitrary number of inner rings. The inner rings have the opposite edge order sequence from the outer ring.

Integrated data. A geographic data set in which all feature data are contained in a single coverage. Opposite of layered data.

Key. In a relational data model, one or more columns (attributes) whose values uniquely identify or can be used to select a row.

Layered data. Feature data thematically separated into separate coverages. Opposite of integrated data.

Left edge. The left edge is the first neighbor of the current edge as one moves counterclockwise around the start node of the current edge.

Left face. The face to the left of an edge in a traverse from the start node to the end node.

Levels of topology. See topology.

Library. A collection of one or more coverages contained within a specified spatial extent, all of which share a single coordinate system. Coverages may be tiled in a library. All tiled coverages in a library must share a common tiling scheme, however.

Library attribute. A property of a library. The library attribute table describes the properties of each library in a database.

Line feature. A geographic entity that defines a linear (one-dimensional) structure; for example, a river, road, or a state boundary.

Line feature class. A collection of line features that maintains a homogeneous set of attributes. Composed of edge primitives.

Line feature table. The implementation of a line feature class in a VPF attribute table.

Lineage information. Information that describes processing tolerances, interpretation rules applied to source materials, and basic production and quality assurance procedures. Lineage information should include all available information from the source.

Logical consistency. Logical consistency refers to the fidelity of the relationships encoded in a data set. In a VPF data set, logical consistency requires that all topological foreign keys match the appropriate primitive, that all attribute foreign keys match the appropriate primitive or features, and that all tables described in the feature class schema tables maintain the relationships described.

Media volumes. As used herein, the number of distinct media that comprise a VPF database. For instance, there may be four CD-ROM media volumes in one database.

Medium. A data storage device (e.g., a CD-ROM, hard disk drive, magnetic tape, or floppy disk).

Metadata. Information about data; more specifically, information about the meaning of other data.

Minimum bounding rectangle. A rectangle of coordinate tuples that defines the minimum and maximum coordinates of an entity. Abbreviated as MBR.

NaN. Stands for not a number. Used as a floating point null value in VPF.

Neutral format. A characteristic of a data model that does not contain product-specific information.

Node. A zero-dimensional geometric primitive that is composed of a single coordinate tuple (pair or triplet). There are two types of nodes: entity nodes and connected nodes.

Outer ring. The outermost boundary of a face, composed of edges ordered in a sequence. The outer ring has the opposite edge order sequence from the inner rings.

Pathname. A file name that uniquely identifies the location path to a file within a series of one or more directories.

Planar model. A planar model is a two-dimensional surface in which every point has a neighborhood (a two-dimensional region) that is topologically equal to a flat disk. It is implemented as a planar graph $\{N, E, F\}$ with a finite number of nodes $N = \{n_1, n_2, \dots\}$, edges $E = \{e_1, e_2, \dots\}$, and faces $F = \{f_1, f_2, \dots\}$ bounded by edges and nodes. Each edge has an orientation from its first (starting) coordinate tuple to its last coordinate tuple. Also, each face of the graph has a certain orientation (cycle) around its edges and nodes. Each edge of a planar model is incident with exactly two faces.

Point feature. A geographic entity that defines a zero-dimensional location; for example, a well or a building.

Point feature class. A collection of point features that maintains a homogeneous set of attributes. Composed of node primitives.

Point feature table. The implementation of a point feature class in a VPF attribute table.

Pointer. A field within a record or within an index that contains the address of a record.

Polygon. Thematically homogenous areas composed of one or more faces.

Positional accuracy. Positional accuracy refers to the root mean square error (RMSE) of the coordinates relative to the position of the real world entity being modeled. Positional accuracy shall be specified without relation to scale and shall contain all errors introduced by source documents, data capture, and data processing.

Primary key. A key whose value uniquely identifies a row.

Primitive. The smallest component of VPF, of which all features are composed. There are three geometric primitives (nodes, edges, faces) and one cartographic primitive (text).

Primitive table. A primitive table inherits the properties of a VPF table, but may also have an associated minimum bounding rectangle table and/or a spatial index file.

Product specification. A document that defines the precise content and format of a specific product. It contains technical requirements and database design decisions such as coding, tiling, special relationships between entities, and so forth. In the context of the VPF, each separate product or application is defined by a product specification and implemented by using VPF structures.

Right edge. The right edge is the first neighbor of the current edge as one moves counterclockwise around the end node of the current edge.

Right face. The face to the right of an edge in a traverse from the start node to the end node.

Ring. A connected set of edges that composes a closed face border. Any single ring is only referenced to and by a single face. If the same set of edges is shared by two different faces, two rings that correspond to the two faces are created from the single edge set. Rings only occur at level 3 topology (when faces are also present).

Row. An ordered collection of fields pertaining to the entity. A tuple in a relation.

Row id. An integer that uniquely identifies each row in a table.

Schema. A description (or picture or diagram) of the structures of a database system.

Schema table. A schema table defines the tables and their relationships within a coverage.

Semantics. The implied meaning of data. Used to define what entities mean with respect to their roles in a system.

Shape line. An ordered set of one or more coordinate tuples that define the placement and shape of a text primitive.

Simple feature class. Consists of a single type of primitive (face, edge, node, or text) and a feature table. There are four subtypes of simple feature classes: point, line, area, and text.

Source. Source information describes the origin or derivation of a single feature, primitive, or attribute. It includes information about processing of the data as well as information about the data source.

Spatial index. A data structure file that allows for the rapid identification of a primitive by using the values of the primitive's coordinates.

Start node. The first node of an edge.

Syntax. The rules governing the construction of a machine language or machine representation of entities.

Table. An organizational structure for data content. In the relational model, a table is a group of repeating rows defined by columns. Equivalent to a relation.

Text feature. A cartographic entity that relates a textual description to a zero- or one-dimensional location. A text feature usually contains information such as font, color, and height.

Text feature class. A collection of text features that maintains a homogenous set of attributes. Composed of text primitives.

Text feature table. The implementation of a text feature class in a VPF attribute table.

Text primitive. Characters placed in specific locations in a coordinate system. Text is a cartographic object, rather than a geographic entity, since it does not participate in topology. A text array indicates a fixed-length string of characters. A text string indicates a variable-length collection of characters.

Thematic attribute. A column in a table that provides a thematic description of a feature. For example, a feature class that contains rivers may have attributes such as width, depth, and name.

Thematic index. A file that allows software to access the row ids of its associated table. In a VPF table, the index is created on a column. Four special indexes are used for feature tables: point, line, area, and text thematic indexes.

Theme. An organizational concept used in the design of spatial databases. Common themes in spatial geographic databases are transportation, hydrology, and soil/land suitability.

Tile. A spatial partition of a coverage that shares the same set of feature classes with the same definitions as the coverage. The topology of each tile is independent of that of each other tile in the coverage.

Tiled coverage. A coverage that has been physically partitioned into tiles.

Tiling scheme. The scheme used to define tile shape and size and to identify tiles (assign identification numbers).

Topology. The branch of mathematics concerned with geometric relationships unaltered by elastic deformation. In geographic applications, topology refers to any relationship between connected geometric primitives that is not altered by continuous transformation. VPF recognizes four levels of topology. Level 0 topology manipulates the purely geometric aspects of the spatial data. No topological information is stored in level 0 topology. Level 1 topology maintains a nonplanar graph. Level 2 topology maintains a planar graph. Level 3 topology explicitly represents the faces defined by the planar graph. See Figure 10.

Traverse. An algorithm that uses winged-edge topology to retrieve a series of neighboring edges to satisfy a query of a network.

Triplet id. A variable-length structure used to contain information for crossing tile boundaries. The first field contains the internal primitive id (referred to as ID). The second field contains the tile reference coverage id (TILE_ID), and the third field contains the primitive id in the associated tile (EXT_ID).

Tuple. See coordinate tuple.

Universe face. The unbounded region surrounding a level 3 topology coverage. The universe face always maintains the first record in a face table.

Variable-length column. A column whose length is determined by the amount of storage needed to store its contents. Useful for character strings and coordinate strings.

Vector. Indicates a collection of coordinate tuples to define a geographic or geometric entity.

Vector product format. A standard format, structure, and organization for large geographic databases based on a georelational data model and intended for direct access. Abbreviated as VPF.

VPF table. A VPF table consists of a table and (optionally) an associated narrative table and/or (optionally) an associated variable-length index file. Primitive tables and attribute tables are derived from VPF tables. Feature tables are derived from attribute tables. Attribute tables may have associated value description tables. Primitive tables may have associated minimum boundary rectangle tables.

Winged-edge topology. A topological construct that connects each edge to two of its neighboring edges, allowing topologic traversal of an edge and/or face network. A neighboring edge is any edge that shares a start or end node with the original edge.

An edge has a start node, which is connected to the left edge, and an end node, which is connected to the right edge.

4. GENERAL REQUIREMENTS

4.1 General. Vector product format is a generic geographic data model designed to be used with any digital geographic data in vector format that can be represented using nodes, edges, and faces. VPF is based upon the georelational data model, combinatorial topology, and set theory (see appendix A for discussions of these concepts).

A VPF-compliant database product must include all mandatory tables and columns which are described in section 5. Similarly, a VPF-compliant application must properly interpret all mandatory and optional tables and columns enumerated in this document.

4.2 VPF characteristics. VPF is a model for digital geographic databases that are intended to allow flexibility in encoding and yet permit direct data access from a variety of applications operating on different computer systems. VPF characteristics are as follows:

- a. Sheetless database support. VPF is designed to support a sheetless database by providing logically continuous topological relationships even when the database itself is physically partitioned into tiles. VPF structures support the query and retrieval of data that extends across tile boundaries.
- b. Neutral format. VPF has a product-neutral format that must be used in combination with an individual product specification to create a product. VPF is topologically structured and supports various levels of topology, from a simple list of coordinates to planar model topology.
- c. Attribute support. VPF uses tables for attribute handling. The tables support both simple and complex features.
- d. Data dictionary. VPF contains a self-defining data dictionary that permits user understanding of features and their attributes.
- e. Text and metadata support. Text information may be encoded either as attributes of features or as 'free floating' text primitives. VPF is compatible with all written languages, including those with accented characters and diacritical marks. In addition to supporting basic cartographic information, VPF supports a variety of metadata files containing information about all or part of the database.
- f. Index file support. Index files that are desirable to enhance database retrieval performance are also

incorporated within VPF. These can include spatial and thematic indexes.

- g. Direct access. VPF allows application software to read data directly from the storage medium without prior conversion to another format. VPF uses tables and indexes that permit direct access by spatial location and thematic content.
- h. Flexible, general-purpose schema. VPF can represent digital geographic data in vector format by providing flexibility in the modeling of any feature data organization, from fully layered to completely integrated. VPF supports coordinate pairs and triplets, multiple scales, and the creation of multiple products.
- i. Data quality. VPF includes standards for data quality reporting and representation. It provides multiple methods for the representation of the spatial and aspatial aspects of data quality.
- j. Feature definitions. VPF organizes features and thematic attributes to allow creation of products that are logically consistent and complete.

4.3 Relationship between VPF and specific products. VPF establishes a standard data model and organization, providing a consistent interface to data content. Data content itself shall be defined in a product specification that determines the content of the feature tables and the relationships between them. VPF can also accommodate additional tables that are not required by VPF itself. Figure 1 illustrates the relationship between VPF and specific products.

4.4 VPF hierarchy. VPF can be viewed as a five-level hierarchy of definitions (figure 2) that increase in degree of abstraction from the bottom up. The bottom two levels define the physical representations of various data structures utilized in VPF. The data structure level concerns the logical representation of VPF data objects. These objects are elements in the VPF data model. The data model describes the data objects and the relationships among them. The top level, which contains the product specification, is used to tailor VPF to the requirements of the product.

This document gives a definition of VPF that encompasses the bottom four levels. A product specification is a combination of the conceptual database design and implementation details required to develop a product that is compliant with VPF. The conceptual modeling of feature classes and coverages is the first responsibility of the author of the product specification. This includes defining the list of features, attributes, attribute values, and providing their definitions. The physical design of

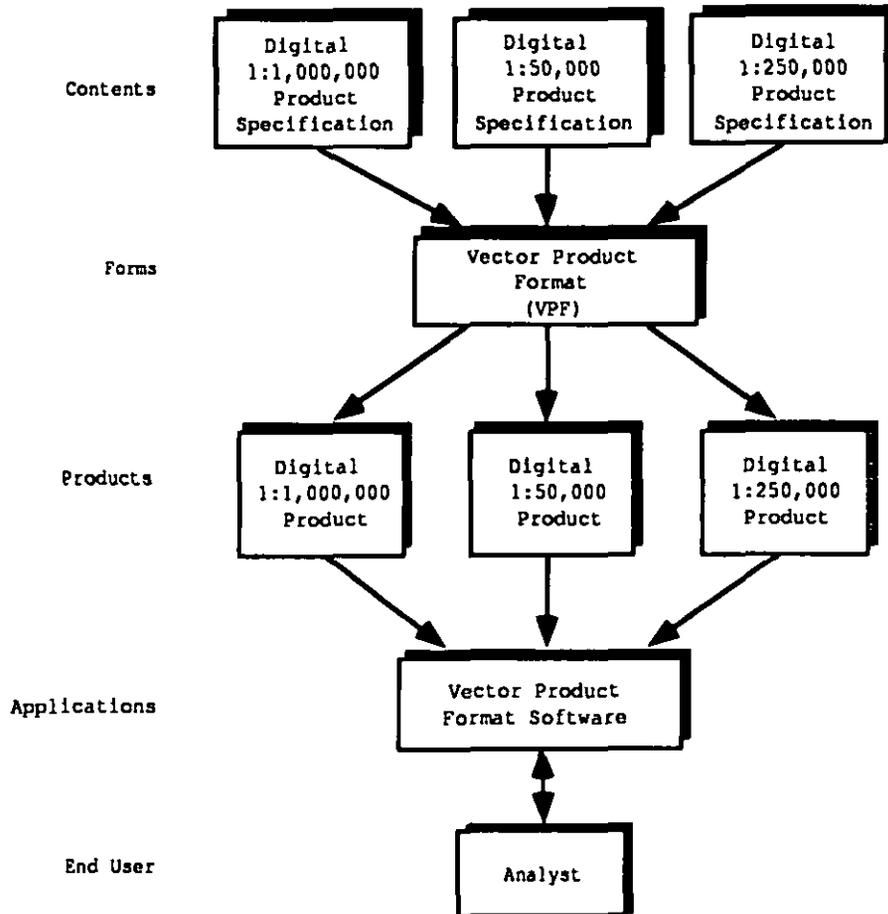


FIGURE 1. Relationship between VPF and specific products.

the product database is also the responsibility of the author of the product specification. Physical design considerations include determining the tiling scheme, the topology level, the feature to/from primitive relationships (i.e., 1:1, 1:N, N:1, and N:M), the column types, and the table definitions.

The first VPF product specification was the product specification for the Digital Chart of the World Database (MIL-D-89009). It may be used as a model by authors of other VPF product specifications.

Vector product format is defined in section 5. Section 5.2 defines the data objects in VPF and the various roles these objects play in the data model. The logical data structures are described in section 5.3, which explains the implementation of VPF; that is, how to construct VPF-compliant databases and applications.

Encapsulation is defined in section 5.4, which identifies the data structure fields. Data syntax is described in section 5.5.

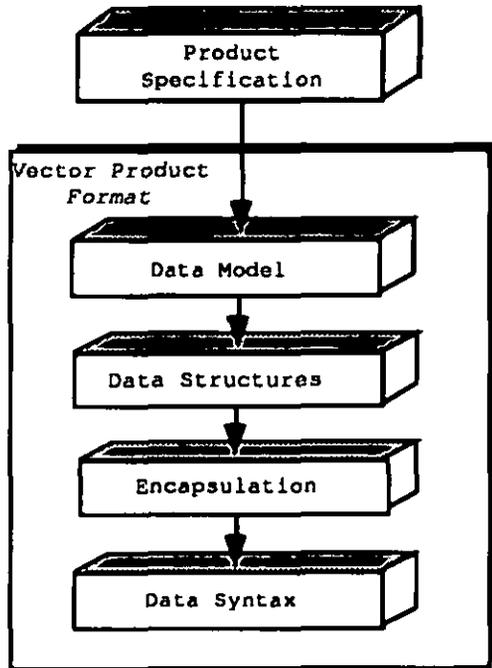


FIGURE 2. Vector product format structure.

5. DETAILED REQUIREMENTS

5.1 General. This section describes the necessary components of vector product format. Section 5.2 discusses the conceptual components of the VPF data model (see appendix A for an overview describing the data model). Section 5.3 contains definitions of the data structures implemented in VPF. The encapsulation of VPF field types, table construction, and indexing structures are discussed in section 5.4. The encoding of the data syntax is found in section 5.5.

5.2 VPF data model. The discussion of the data model is broken into three subsections: data organization, VPF data model components, and data quality. Each of these subsections addresses the data model from a different perspective. The data organization subsection (5.2.1) addresses VPF by defining the physical structures that make it up. Only three structures are used to implement the entire VPF data model: directories, tables, and indexes. The data model component subsection (5.2.2) addresses VPF by defining the entities in a geographic database and describing the way in which these entities are captured through the physical VPF structures, starting with the most basic components (primitives) and continuing through the other levels (features, coverages, libraries, and database). Finally, the data quality subsection (5.2.3) describes the options available in VPF for the maintenance of data quality information at any level in the data model.

5.2.1 Data organization. All VPF data are organized in the form of files. A file is a named, sequentially ordered stream of bytes (figure 3). Files may be created, deleted, opened, closed, read (from byte m to byte n), and written (from byte m to byte n).

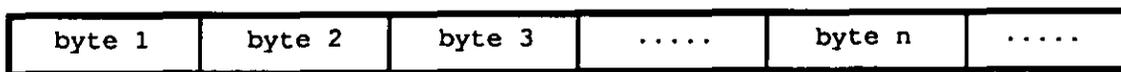


FIGURE 3. Byte stream.

VPF uses only three types of files: directories, tables, and indexes.

5.2.1.1 Directory. The directory is a file that identifies the names of a collection of files, their types (directory or table), and their beginning addresses and lengths (table 1).

TABLE 1. Directory structure.

Directory			
Name	Type	Address	Length
File Name	Directory or Table File	Location on Medium	Length In Media Storage Units

VPF directories are strictly hierarchical; each file is contained in exactly one directory. File names must be unique within a directory. A file referenced by a directory is said to be contained in that directory. A file contained in a directory may be referenced by a special form of its name called a pathname (because it contains the location path to the file). A pathname has the following form:

<directory name><separator><file name>

where:

< > indicates that the enclosed name element is to be replaced with the actual text string indicated.

VPF uses the backslash character (\) as the generic pathname directory separator. For platforms requiring a different separator, software will replace the backslash with the appropriate separator character. For example, if a file named ROADS.LFT is contained in a directory named URBANAREAS, and the separator is (\), the resulting pathname would be:

URBANAREAS\ROADS.LFT

Directories are themselves files, so they may be contained in other directories. They are referenced by pathname the same way as other files. Thus, if URBANAREAS is contained in LIBRARY1, the resulting pathname would be:

LIBRARY1\URBANAREAS

Finally, pathnames may be combined. Since the directory that contains a file can be contained within a directory itself, it is necessary to have a form of file name that uniquely identifies that file contained within that directory (file names, while unique within a directory, are not unique between directories). For our example, that form would be:

LIBRARY1\URBANAREAS\ROADS.LFT

5.2.1.2 Tables. In the VPF data model, the table is the organizational structure for all data content. All tables in a VPF database share a common basic structure; this structure, which is described in the VPF table components section below (5.2.1.3), is mandatory for all VPF tables.

By definition, a VPF table must include at least the basic structure. Optionally, a VPF table can also include two additional structures: the narrative table and the variable-length index file. In the VPF data model, all geographic phenomena are modeled by VPF tables or by tables derived from a VPF table. A table derived from a VPF table is one that possesses all the properties of a VPF table but also has additional properties that support other specific functions.

The primitive table and the attribute table are examples of derived VPF tables. A derived table can also be further specialized to satisfy a particular need. A feature table may be derived from the attribute table, for example. Figure 4 graphically shows a VPF table and three derived tables. The VPF table is drawn with an oval; all derived tables are differentiated from VPF tables and from each other by being drawn with other distinct shapes.

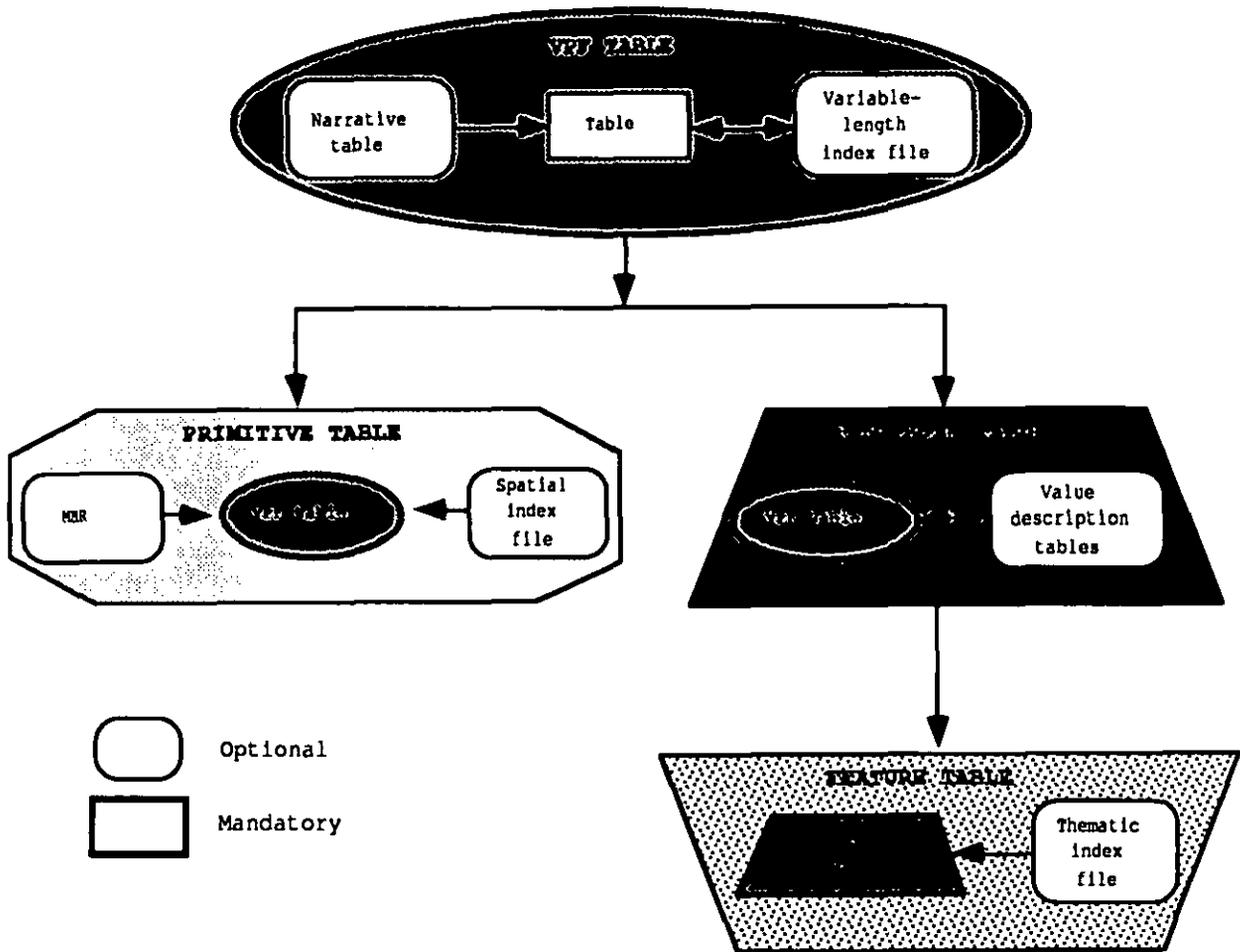


FIGURE 4. Table types. MBR denotes minimum bounding rectangle.

As the figure shows, a VPF table may have an associated index file for variable-length records and a narrative table. A primitive table (discussed in section 5.2.2.1) may possess these two tables associated with its VPF table, but may also have a spatial index file and a minimum bounding rectangle table. An attribute table may also possess the tables associated with a VPF table, but may also have value description tables that provide the data dictionary for the table. The same data dictionary table may be shared by more than one attribute table. Finally, a feature table inherits the tables associated with an attribute table, but may also have a thematic index file. Feature tables are discussed further in section 5.2.2.2.

5.2.1.3 VPF table components. VPF tables consist of the following parts: a table header, a row identifier, and the table contents. The table header contains the metadata about a table and the column definitions. Columns are defined by a name and a data type; each column must have a name that is unique within the table.

Data contents in VPF tables are organized into rows and columns. All rows in a table share the same column definitions. Each row in the table is defined by a row identifier (row id). Table 2 depicts the principal components of a VPF table.

TABLE 2. VPF table structure.

Table Header	
Metadata and column definitions:	
<ul style="list-style-type: none"> a. Table description b. Narrative file name (optional) c. Column definitions: <ul style="list-style-type: none"> Column name Field type Field length Key type Column textual description Optional value description table name Optional thematic index file name Optional column narrative file name 	
ID	Table Contents
Indicates the starting position of each row.	The data composing the table that match the column definitions.

This document describes the column definitions for all the VPF standard-specified columns, and the table organization for those columns. Data columns and tables described in this document are labeled either mandatory or optional. A VPF product must include all mandatory tables and columns. It is not possible to remove any mandatory column from any table. A VPF-compliant application

must be able to process a VPF product and interpret all mandatory and optional columns as described in this document.

Additional product-specific columns are allowed by VPF. If present, these columns must be defined in their product specifications. Product-specific columns must not alter the use of the columns specified in this document.

5.2.1.4 Indexes. A table may have associated indexes. If a table contains a variable-length coordinate string column or a variable-length text string column, a separate index file must be present.

In addition to variable-length indexes, VPF also supports spatial and thematic indexes. Spatial indexes contain references to row data that are based on the value of a coordinate column. Thematic indexes contain references to row data that are based on the value of noncoordinate columns.

5.2.1.5 Narrative tables. Each VPF table may have an associated narrative table that provides miscellaneous information about the VPF table. The purpose of the narrative table is to provide the database designer with the ability to record comments or information pertinent to the associated table. The narrative table name is stored in the VPF table's header information. See section 5.3.8 for more details.

5.2.1.6 Attribute tables. Real-world objects are referred to as entities or features; they are modeled as tables in VPF. The properties of entities are called attributes. In an attribute table, one table column is defined for each attribute describing an object. Each object occupies a row in the table. Examples of attributes include data quality, size, and name. A sample attribute table is shown in table 3.

A column or a group of columns that can be used to identify or select a row is called a key. A unique key is a key that uniquely identifies each row. One unique key is designated the primary key; each table has one and only one primary key. In the city attribute table (table 3), the Built-Up Area column is the primary key.

A relational join is a database operation that brings together a number of tables into a new relation by using a set of common keys. The tables in such joins are called base tables. When a common key in a join is the primary key in one of the base tables but not in another, the non-primary (yet common) key is called a foreign key. In the city attribute table (table 3), the State column is a foreign key; in the state attribute table (table 4),

TABLE 3. City attribute table.

ID	Built-Up Area	State	Population Size	Median Income per Household
<i>implicit</i>	<i>character string</i>	<i>character string</i>	<i>binary integer</i>	<i>binary integer</i>
UNIQUE KEY	PRIMARY KEY	NON-UNIQUE	NON-UNIQUE	NON-UNIQUE
1	Los Angeles	California	2966850	15735
2	New York	New York	7071639	13854
3	Salt Lake City	Utah	163033	13211
4	Las Vegas	Nevada	164674	17468
5	San Francisco	California	1366383	16782

TABLE 4. State attribute table.

ID	State	Area (sq. mi.)	Total Population
<i>implicit</i>	<i>character string</i>	<i>binary integer</i>	<i>binary integer</i>
UNIQUE KEY	PRIMARY KEY	NON-UNIQUE	NON-UNIQUE
1	California	158706	26365000
2	Nevada	110561	936000
3	New York	49108	17783000
4	Utah	84899	1645000

the State column is the primary key. In the city attribute table (table 3) the State column becomes a foreign key only through its reference by the state attribute table (table 4).

5.2.2 VPF data model components. The VPF data model may be considered to be layered into five structural levels (figure 5). At the lowest level, a VPF database consists of geometric and cartographic primitives. These primitives define the spatial aspects of entities. Primitives that are accompanied by thematic information that helps give them meaning are called features. Both features and primitives make up coverages, which in turn make up libraries; and finally, a database is made up of libraries.

An analogy can be drawn between VPF and written language. Letters are at the bottom of the language hierarchy. Words are made up of letters. In turn, sentences are made up of words. An essay is created from sentences, and a collection is made up of essays. Each of these entities has a distinct and different meaning not

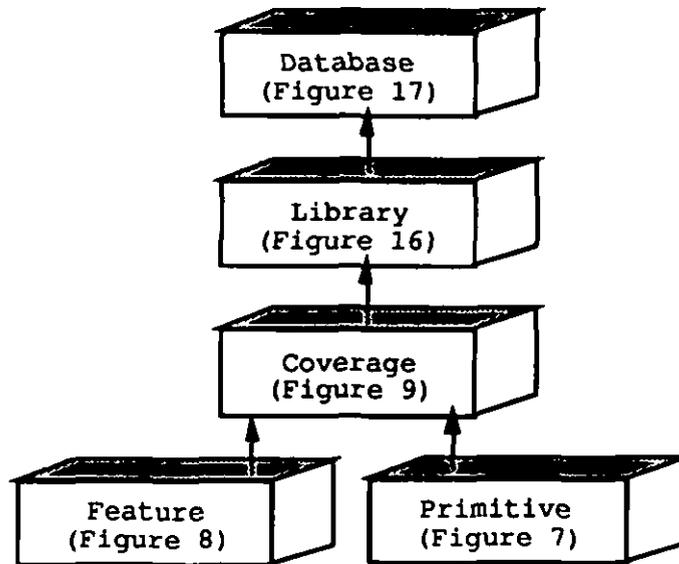


FIGURE 5. VPF structural levels.

possessed by the entities below. The content of each entity, however, depends on that of the constituent entities. Databases and libraries are used primarily to facilitate data access, whereas coverages (which incorporate topology) are used to define the relationships between features.

Figure 5 depicts the VPF hierarchy. Of the figures it cites, all except figure 8 can be referenced to determine the required and optional components of a VPF database.

5.2.2.1 Primitives. There are three geometric primitives in VPF: nodes, edges, and faces (figure 6). As figure 6 shows, there are two types of node primitives: entity nodes and connected nodes. There is one type of cartographic primitive, text. These four primitives are combined to model any geographic phenomena using vector geometry. All primitives except text can be linked to each other by topological relationships, which are discussed further in section 5.2.2.3.1.

The following sections summarize each of the primitives. Figure 7 depicts each primitive and its associated tables and indexes.

5.2.2.1.1 Nodes. Nodes are zero-dimensional primitives that are used to store significant locations. There are two types of nodes: entity nodes and connected nodes.

- a. Entity nodes. Entity nodes are used to represent isolated features that are either truly zero dimensional, such as survey points, or too small to resolve at the

collection scale, such as water towers at 1:24,000 scale. An entity node is topologically linked to its containing face when face topology is present. Entity nodes can reside at any location, whether or not there is another primitive at that same location.

- b. **Connected nodes.** Connected nodes are always found at the ends of edges and are topologically linked to the edges. Connected nodes are used in two ways: (1) to define edges topologically (always) and (2) to represent point features that are found at a juncture of linear features, such as overpasses, locks in a canal, or underground utility access points. Under the first usage, the connected nodes are referred to as start and end nodes. Under the second usage, attributes will be associated with the point features related to the connected nodes. If many edges intersect a node, only one edge will be maintained per node in the connected node table; other edges are linked by using winged-edge topology.

5.2.2.1.2 **Edges.** Edges are one-dimensional primitives that are used to represent the locations of linear features (such as roads) and the borders of faces. Edges are composed of an ordered collection of two or more coordinate tuples (pairs or triplets). At least two of the coordinate tuples must be distinct. The orientation of an edge can be recognized by the ordering of the coordinate tuples.

Edges are topologically defined by nodes at ends (level 2 topology); edges, in turn, define faces (level 3 topology). In addition to the Start Node and End Node columns, the edge primitive table contains column information (Right Edge, Left Edge, Right Face, Left Face) that is necessary to support higher levels of topology. This topology information permits the query and retrieval of features. The direction of an edge is its orientation from start node to end node. A table is maintained for each edge primitive; the table contains the minimum bounding rectangle for the edge. Appendix B describes the use of winged-edge topology, which is used with edge primitives.

5.2.2.1.3 **Faces.** A face is a two-dimensional primitive enclosed by edges; faces are used to represent area features, such as countries, inland water, or urban areas. Faces are defined by topological references to a set of edges that compose the face border. A face may have interior borders as well as exterior borders, allowing for faces that have other smaller faces within them. This relation consists of a reference to the start of a closed ring of edges, which may then be followed clockwise to close the ring. A face may consist of multiple rings; there may be one outer ring and zero or more inner rings. Faces are non-overlapping, and the faces in a coverage completely exhaust the area of a plane. A minimum bounding rectangle table is maintained for each face primitive.

5.2.2.1.4 Text. Text is a cartographic rather than a geometric object. Text strings can be placed in specific locations in geographic space. Text can be used to associate names with regions that are vague or ill defined, such as the Rocky Mountains. A text primitive may also be used when the name of a feature needs to be located in a specific relationship to a feature and could not otherwise be reproduced. For example, the text "Pacific Ocean" may be required for graphic display on a map, and may therefore be encoded as a text string, even though it is also being stored as an attribute of a face in a hydrographic coverage. Text primitives do not participate in topology.

5.2.2.2 Feature classes. Features are defined using primitive and attribute tables by means of relational modeling. Tables are related to each other by their common keys. The relationships between tables are determined by the product specification.

5.2.2.2.1 Feature definition. A feature is represented by a set of one or more primitives, a single row of attribute data in a feature table which uniquely identifies the feature, and zero or more rows of attribute data in other tables. A simple feature (e.g., a building) may consist of only a single face and a single row of attribute data. A complex feature (e.g., an airport) will be identified by one row in a complex feature table, but will include the additional information contained in other feature tables.

Features are grouped into feature classes. Each feature class is individually defined by a set of attributes (column definitions) and is uniquely named. The rows of features in a feature class collectively form the feature table for the feature class. Every feature class has one and only one feature table. The feature table is a special form of an attribute table because it directly references a feature. Table 5 expresses the basic structure of a feature table in VPF.

TABLE 5. Feature table structure.

Primary Key	Attributes
Either a primitive row identifier or feature definition table id (may be the table id).	Attributes as specified in the product specification, or join values for reference into other attribute tables.

5.2.2.2.2 Feature table joins. Simple features may be composed of one or more primitives of a single type, while complex features may be composed of one or more simple or complex

features. A feature join designates which primitives belong to which features. Four types of feature joins represent all the possible relationships between features and primitives: one-to-one, many-to-one, one-to-many, and many-to-many. Appendix C provides a detailed discussion of these four types of join columns and feature join tables.

5.2.2.2.3 Feature class types. There are three types of feature classes in VPF: simple feature classes, complex feature classes, and text feature classes. Figure 8 portrays the structural schema of these feature classes.

- a. Simple feature classes. A simple feature class consists of a (logically) single primitive table and a single simple feature table. There are three types of simple feature classes in VPF:
 1. Point feature classes (composed of entity or connected nodes)
 2. Line feature classes (composed of edges)
 3. Area feature classes (composed of faces)
- b. Complex feature classes. A complex feature class consists of one or more simple feature classes, one or more complex feature classes, or both, and a single complex feature table, all within one coverage. For example, a complex watershed feature may be constructed from simple features, such as rivers, springs, and lakes.
- c. Text feature classes. A text feature class consists of a text primitive table and a text feature table. The text feature class is not a true feature class, but it is often useful to process text as if it were a feature. For instance, many maps contain text annotation that does not reference a specific geographic entity. The text "Himalaya Mountains" may not define any geometric primitive or feature, but merely provide associative information for the viewer. Using a text feature allows thematic queries on text just like other features. For instance, if a text feature has a height attribute, software can retrieve 'all text with HEIGHT > 0.5'.

5.2.2.2.4 Constructing feature classes. A feature class consists of a set of tables that includes at least one primitive table and one feature table. The rules for constructing feature classes are stored in the feature class schema table, which describes how each table relates to each other table in the feature class (table 6).

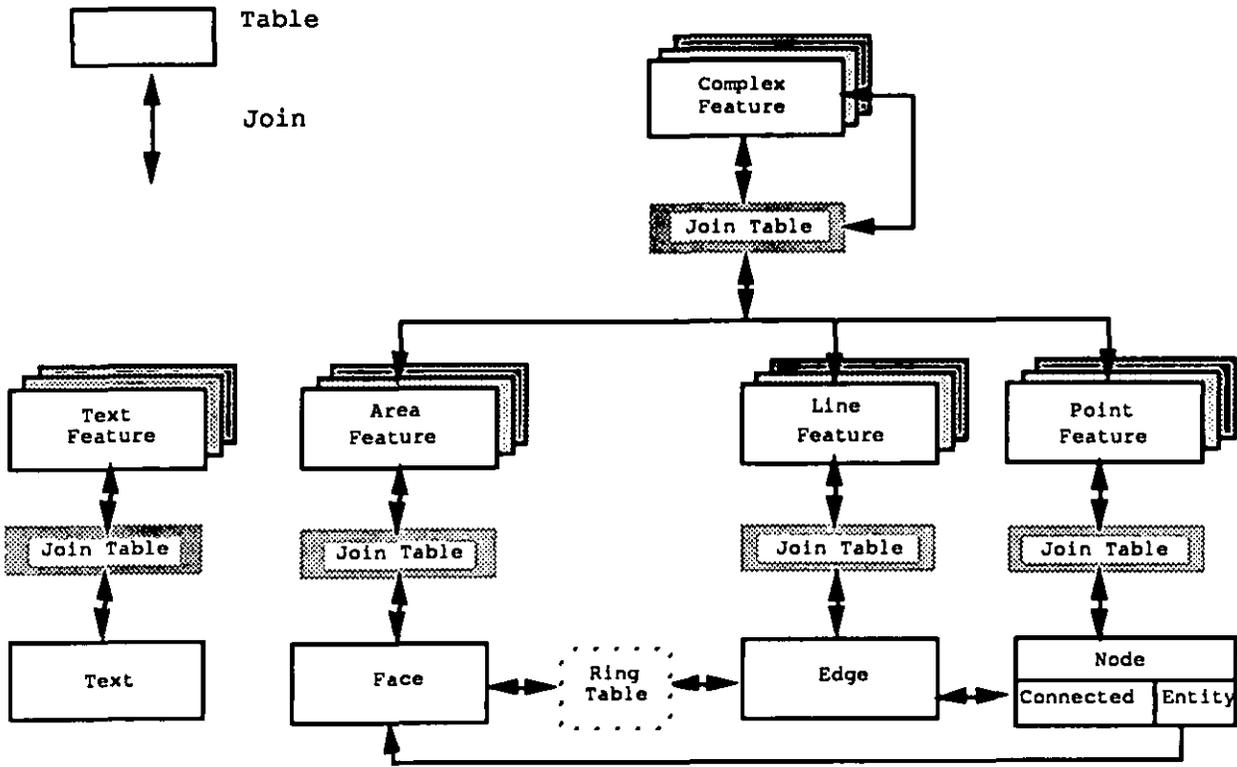


FIGURE 8. Feature class structural schema.

TABLE 6. Feature class schema table.

Column Name	Description
ID	Required row id
FEATURE_CLASS	Name of the feature class
TABLE1	The first table name in the relationship
TABLE1_KEY	Column name of table 1 join key
TABLE2	The second table name in the relationship
TABLE2_KEY	Column name of table 2 join key

Table 7 shows a feature class schema table and an example of a simple feature class. Within the schema table, the feature class is named TRNLINE. The first table in the relation is called TRNLINE.LFT. The second table is named EDG, which is the standard label for the edge primitive. The key column, ID, in TRNLINE.LFT relates to the key column, ID, in EDG. TRNLINE.LFT has six attributes: F_CODE, BOT, LEN, LMC, TUC, and FROM_TO. F_CODE, BOT, LEN, LMC, and TUC are all feature attribute coding system (FACS) codes for a feature. FROM_TO, on the other hand, describes the geometry of the feature. A FROM_TO value of 1 means the feature has the same orientation as its related primitive; a -1 means opposite orientation. The edge primitive contains the required columns for level 2 topology (see section 5.2.2.3.1).

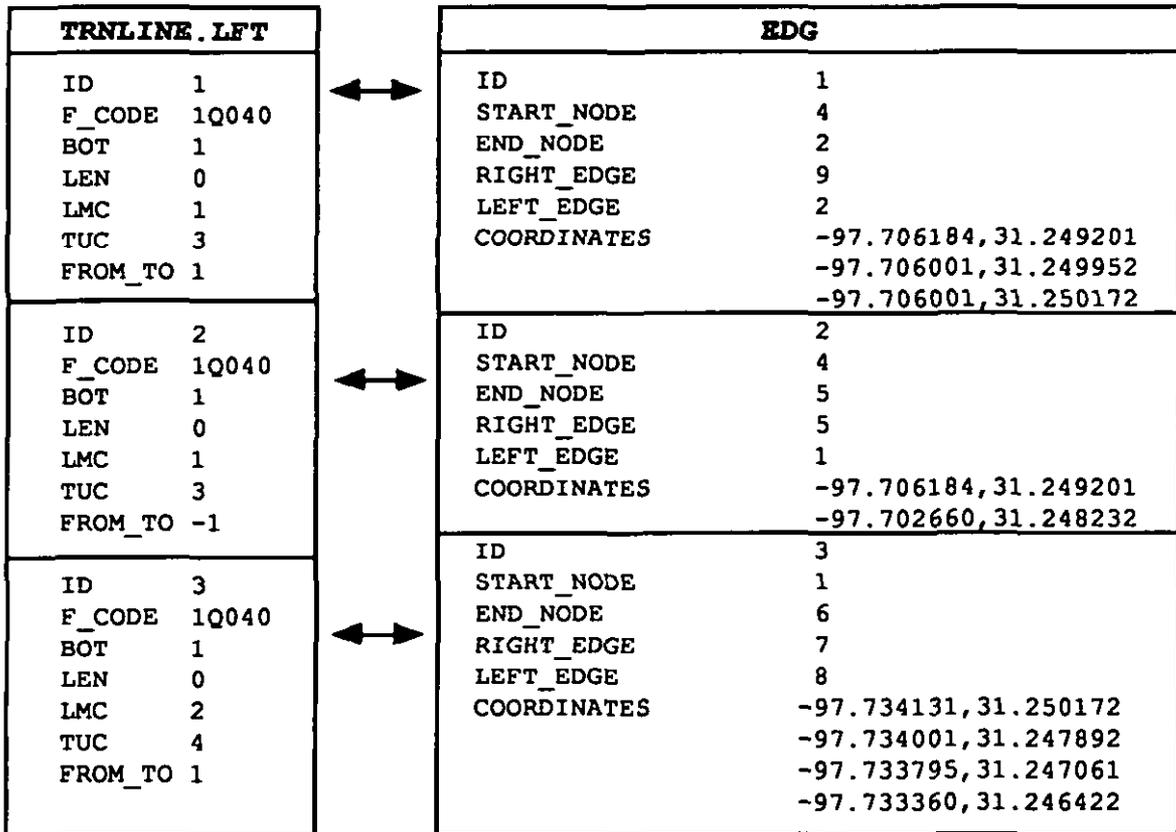
Appendix C provides additional information on feature classes and feature joins.

TABLE 7. Feature class schema table and simple feature class.

Feature class schema table

ID	1
FEATURE_CLASS	TRNLINE
TABLE1	TRNLINE.LFT
TABLE1_KEY	ID
TABLE2	EDG
TABLE2_KEY	ID

Simple feature class



5.2.2.3 Coverage. A coverage is composed of features whose primitives maintain topological relationships according to a level of topology (level 0, 1, 2, or 3) defined for the coverage. All of the file structures that make up a coverage are stored in a single directory. A coverage is generally analogous to a photographic separate in conventional cartography.

At the coverage level (see figure 9), there are four mandatory components: the tile and primitive directories, the feature tables, the value description tables, and the feature class schema table. A variable-length index file is mandatory whenever a variable-length column appears in the feature tables. Maintaining a data quality table at the coverage level is optional.

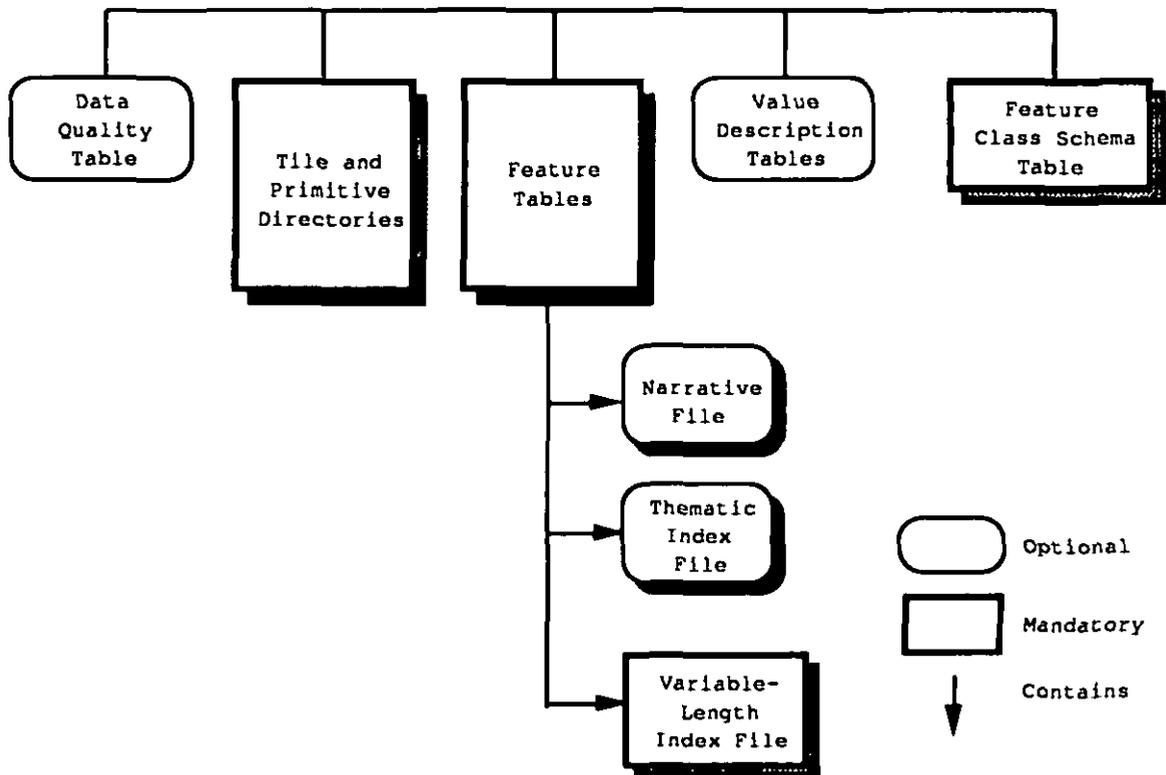


FIGURE 9. Coverage contents.

5.2.2.3.1 VPF topology. There are four recognized levels of topology in VPF coverages, ranging from level 3, where all topological connections are explicitly present, to level 0, where no topological information is explicitly present. Figure 10 summarizes the characteristics of these levels and gives an example of each. Since text does not have any topological relationships, it is not listed in figure 10. Text may be included with other primitives at any topological level, even though it does not have any topology.

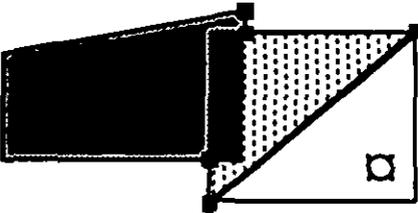
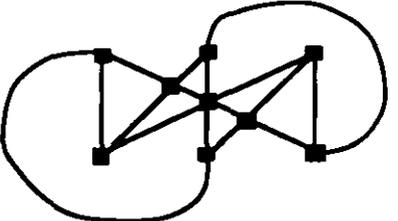
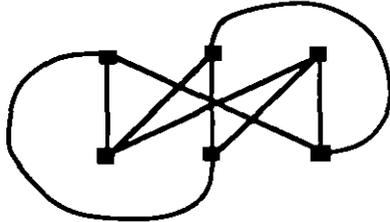
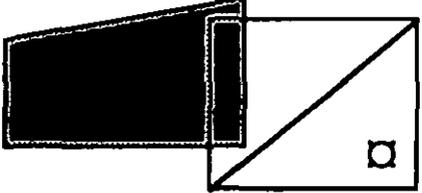
Level	Name	Primitives	Description	Example
3	Full topology	Connected nodes, entity nodes, edges, and faces	The surface is partitioned by a set of mutually exclusive and collectively exhaustive faces. Edges meet only at nodes.	
2	Planar graph	Entity nodes, connected nodes, and edges	A set of edges and nodes where, when projected onto a planar surface, the edges meet only at nodes.	
1	Non-planar graph	Entity nodes, connected nodes, and edges	A set of entity nodes and edges that may meet at nodes.	
0	Boundary representation (spaghetti)	Entity nodes and edges	A set of entity nodes and edges. Edges contain only coordinates.	

FIGURE 10. Levels of topology in VPF coverages.

The columns carried in the edge and node tables, which determine connectivity and adjacency for the topology, depend on the level of topology. For instance, the edge table in table 7 does not contain the level 3 topology columns RIGHT_FACE and LEFT_FACE, because faces do not exist in level 2 topology. Table 8 shows the columns that are mandatory in each primitive table for the required level of topology. The characteristics of these columns are specified in the primitive definitions found in section 5.3.

Figures 11, 12, and 13 use entity relationship (ER) diagrams to portray the primitives and their relationships for each level of topology.

TABLE 8. Columns required to define topology in VPF coverages.

Level	Primitive	Mandatory Columns
3	Face	RING_PTR
3	Ring Table	FACE_ID, START_EDGE
3	Edge	START_NODE, END_NODE, RIGHT_FACE, LEFT_FACE, RIGHT_EDGE, LEFT_EDGE
3	Entity Node	CONTAINING_FACE
3-1	Connected Node	FIRST_EDGE
2-1	Edge	START_NODE, END_NODE, RIGHT_EDGE, LEFT_EDGE
2-0	Entity Node	(none)
0	Edge	(none)

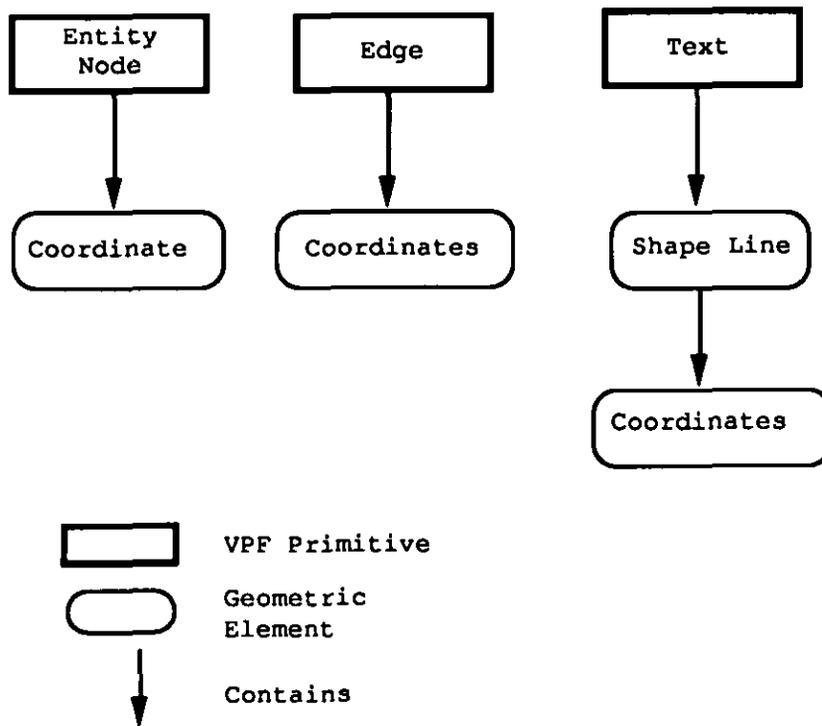


FIGURE 11. Level 0 topological primitives and their attributes.

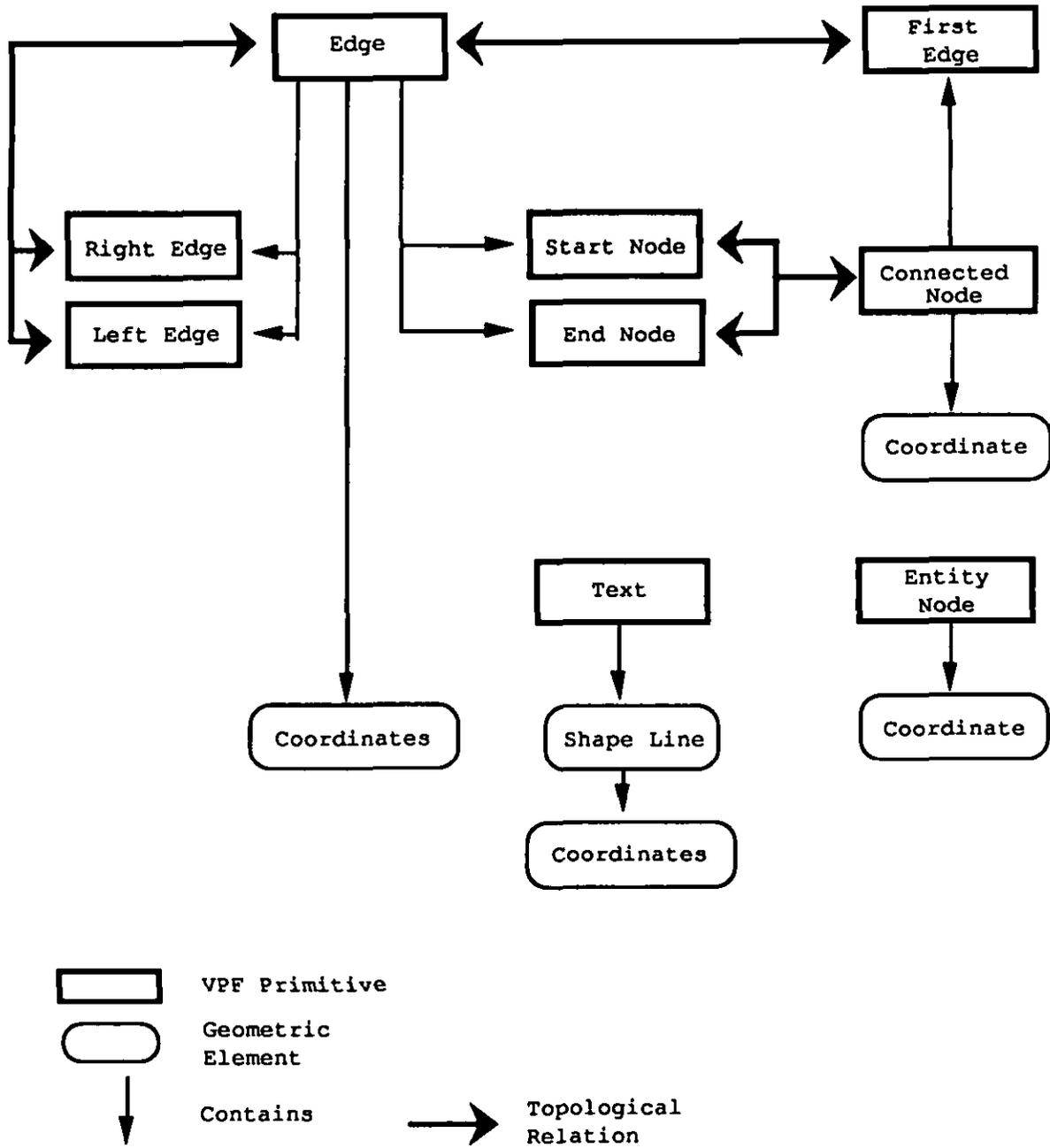


FIGURE 12. Level 1 and Level 2 topological primitives and their attributes.

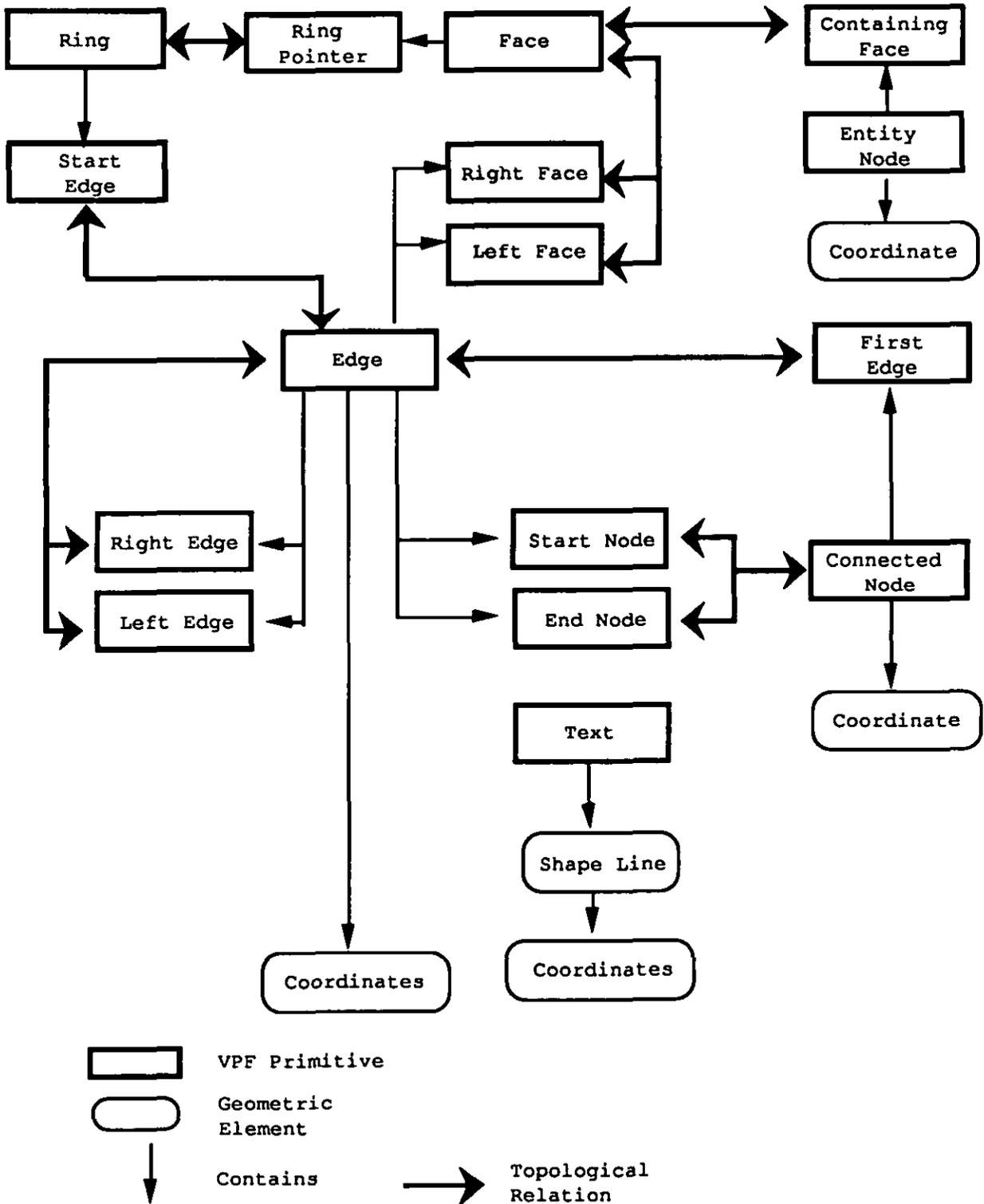


FIGURE 13. Level 3 topological primitives and their attributes.

5.2.2.3.2 Value description tables. A value description table (VDT) is provided to describe coded attributes. There are three types of attribute values: distinct values, integer value codes, and character value codes.

- a. Distinct values. Distinct values are attribute values that can be directly interpreted. Measurements of length or elevation are examples of distinct values. The interpretation of distinct values does not require a value description table.
- b. Integer value codes. In many cases, the values entered in an attribute column are only codes designed to facilitate data processing and transmission. Codes made up of numerals and their corresponding descriptions are maintained in the integer VDT. Both the long and short integer data types share the same integer VDT.
- c. Character value codes. For alphanumeric codes, there is a character VDT similar to the integer VDT. For instance, feature attribute coding systems generally use a five-character-string feature coding scheme.

5.2.2.3.3 Tiled coverages. Tiling is geographically subdividing a coverage solely for the purpose of enhancing data management; a coverage subdivided in such a manner is then referred to as a tiled coverage. A tiled coverage contains the same attribute information as an untiled coverage. The logical interpretation of a tiled coverage is identical to that of an untiled one.

A tiled coverage is physically subdivided into tiles according to a tiling scheme. The tiling scheme (tile boundaries and size of tiles) and the handling of the features that lie on tile boundaries and text primitives that cross borders are all defined by a product specification. Each tile in a tiling scheme has a unique tile identifier. Figure 14 shows a tiling scheme that uses regular rectangular tiles. Appendix D contains more information on tiling.

Each tile can be treated as a single coverage, except that tiles do not contain feature attribute or schema tables. These tables belong to the tiled coverage as a whole.

For database administration purposes, tiled coverages are often distributed to different storage media. For example, 5 tiles of a 10-tile coverage may be sent to one site and the remaining 5 to another. To save storage space under such circumstances, the feature attribute tables should contain only entries for those features that lie wholly or partially in that particular set of tiles.

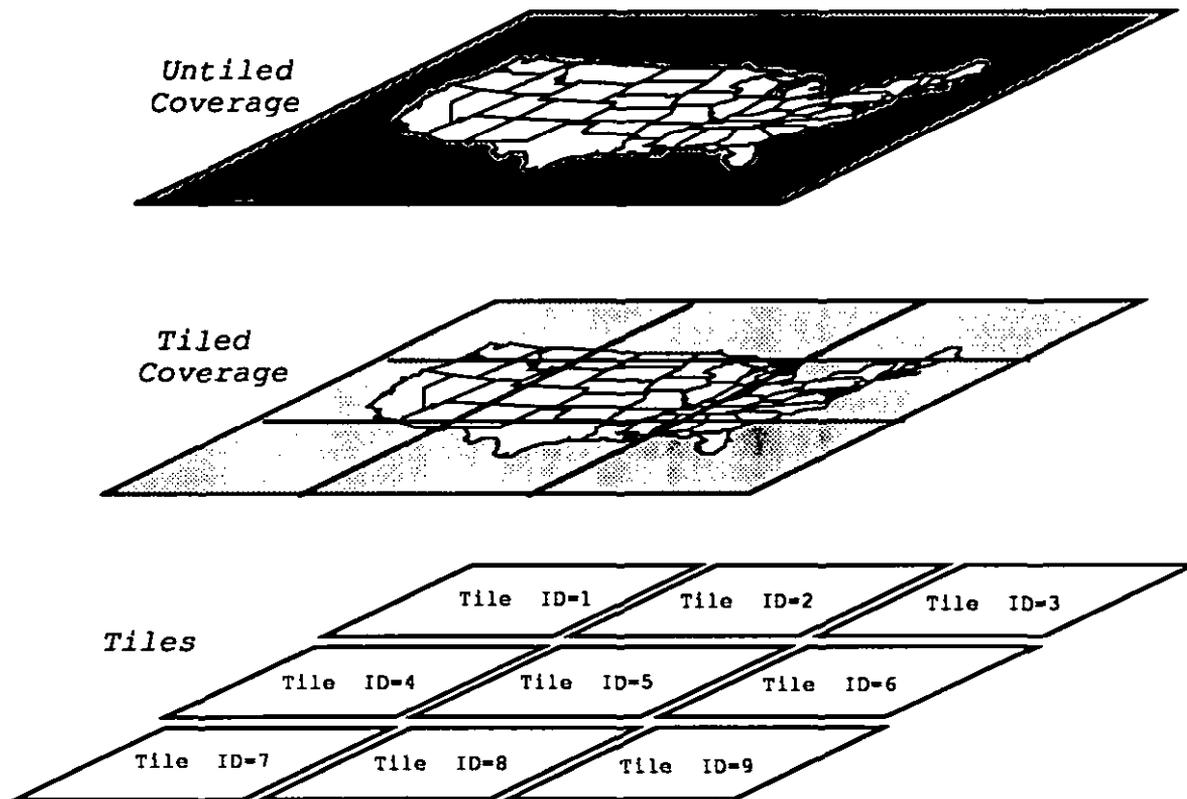


FIGURE 14. A tiling scheme.

However, if each tile is sent separately as an independent coverage rather than as an element of the original tiled coverage, each must contain its own copy of the feature tables, and other tables required to maintain a VPF-compliant database.

5.2.2.3.4 Cross-tile keys. VPF provides a mechanism for maintaining geographic features in a logically continuous spatial database, whether or not a tiling scheme is present. Since the primitives in each tile of a tiled coverage are managed separately from those in other tiles, labels given to primitives are unique only within a tile. In order to support a logically continuous spatial database, a triplet id can be used instead of an integer key to reference primitives across multiple tiles. The triplet id augments the key of a primitive with the key of the tile in which the primitive falls. Appendix B contains a discussion that fully describes this concept.

- a. In a feature table it is necessary to maintain a column that identifies the tile id and primitive id that are related to the feature. This column is named after the primitive table it relates to, followed by '_ID.' Alternatively, the triplet id can be used to reference primitives from multiple files. The first field is left null; the second field in the triplet id is the tile id

(found in the tile reference coverage); and the third field is the primitive row id in that tile.

- b. For an edge primitive, the triplet id is used to maintain cross-tile topology. The Left Face, Right Face, Left Edge, and Right Edge columns are defined as triplet ids to support tiled coverages. The triplet id contains a reference to the internal topology within the current tile; the two other components reference the external tile directory and the edge within that tile. For example, for a face divided by a tile boundary, the external id portion of the Left Face field in figure 15 would include the continuing face in the other tile. This inclusion of internal and external tile references allows software to detect tile borders and continue operations across boundaries or to operate only within the current tile. If a coverage is untiled, the Left Face, Right Face, Left Edge, and Right Edge columns may be defined as integer columns; otherwise the external tile id and primitive id columns of the triplet id will be null.

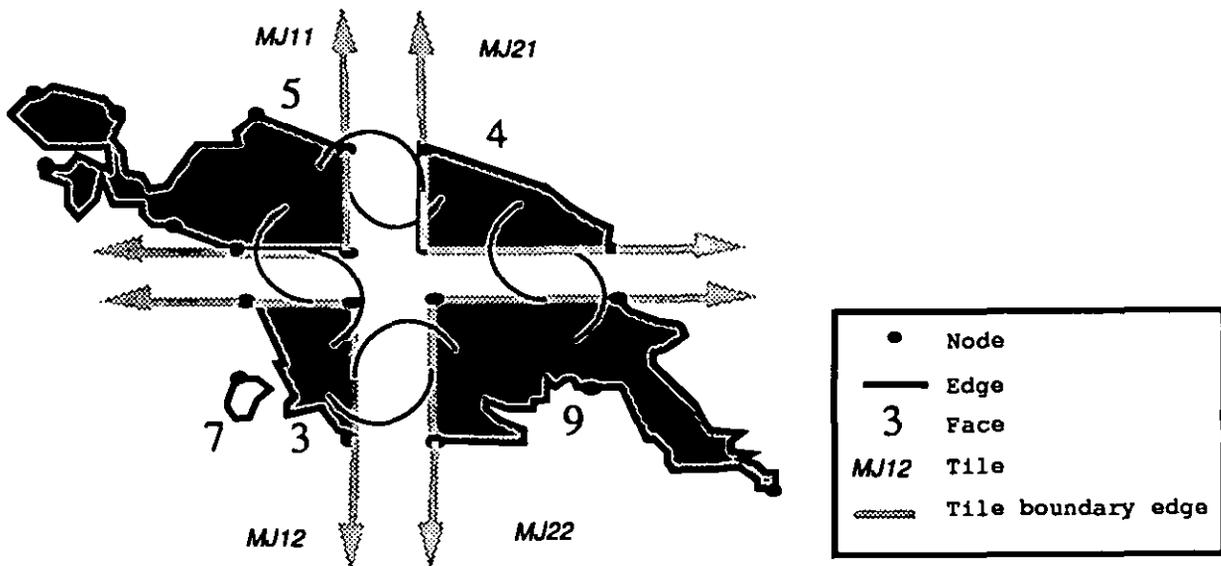


FIGURE 15. Face cross-tile matching.

5.2.2.4 Library. A library is a collection of coverages that share a single coordinate system and scale, have a common thematic definition, and are contained within a specified spatial extent. If any of the coverages composing the library are tiled, then all other coverages must either use the same tiling scheme, or be untiled. The contents and organization of the libraries are determined by a product specification. All of the tables and

coverages making up the library are contained within a single master directory (figure 16).

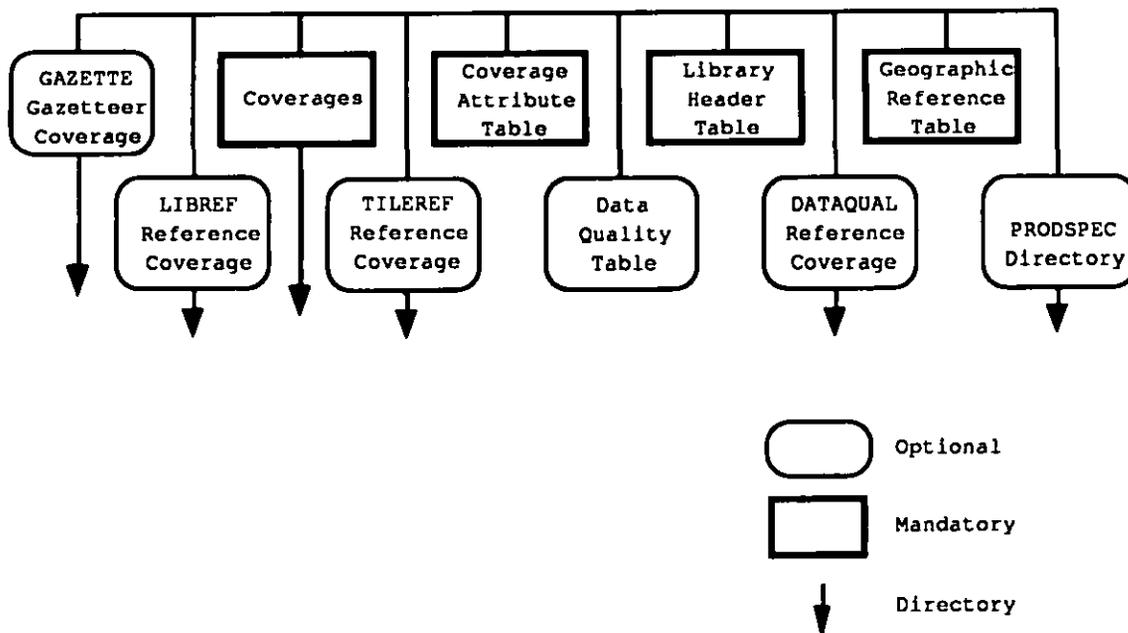


FIGURE 16. Library directory.

5.2.2.4.1 Tile reference coverage. A tile reference coverage is mandatory if a library contains tiled coverages. The spatial extent of the library and its tiling scheme are represented in the tile reference coverage. The reference coverage contains a set of faces identifying the tiles that the library uses to subdivide the region of interest. The universe face always has a face id that equals one. The minimum bounding rectangle that includes all the faces, except the universe face, defines the spatial extent of the library. The tile reference coverage is a standard untiled coverage with level 3 topology.

5.2.2.4.2 Library attributes. General information about a library is stored in the library header table. There is one primary attribute row per library, and zero or more other attribute rows. Additional metadata that is product specific may either be stored in VPF tables in a product-specific directory named PRODSPEC, or additional columns may be added to the metadata tables. Libraries are also the level at which coverage attribute tables reside. The library header table provides a brief description of the coverages in the library. The coverage attribute table also acts as a topics list for a library.

5.2.2.4.3 Library coordinate system. The coordinate system of a library is defined by a geographic reference table. This table defines the basic coordinate system for the library.

Extensions to the basic coordinate system may be provided by a product specification.

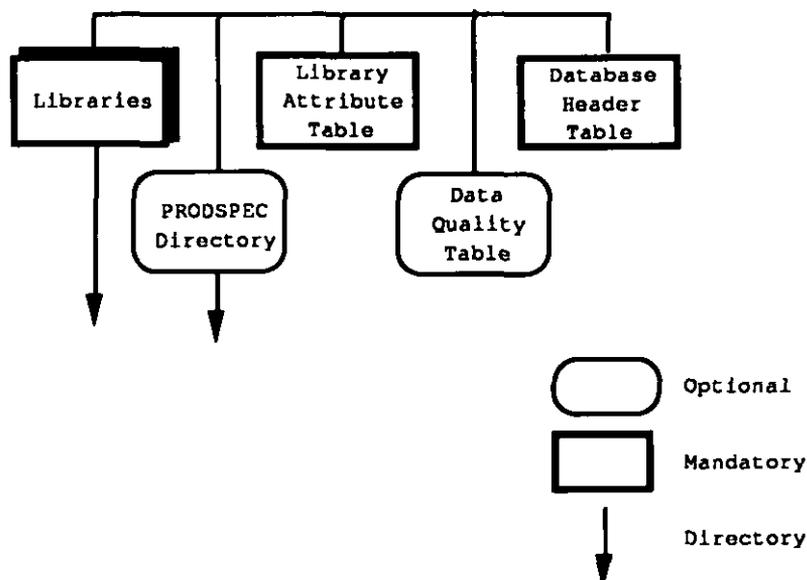
An example of a geographic reference table would document the projection used, its base parameters, and the values used to define the size of the Earth. This information (values for the semi-major and the semi-minor axis of an ellipsoid, other projection datum information, the false origin of a projection, and so forth) is necessary to understand a coordinate system in a VPF library.

5.2.2.4.4 Library reference coverage. When tiles exist in the library, a library reference coverage must exist. This coverage is spatially registered to the tile reference coverage and contains only line features. The purpose of the library reference coverage is to provide software with a preliminary view of the data contained within the library to use for such functions as "zoom out." The contents of this coverage will be a generalized map of the coverage considered to be most significant to the library. For example, if a library contains the rivers, transportation, and political boundaries of the Australian continent, a generalized map of the political boundaries might be considered appropriate for the library reference coverage.

5.2.2.4.5 Data quality reference coverage. It is possible to include a data quality coverage at the library level. This coverage is spatially registered to the tile reference coverage. Its purpose is to record data quality information that pertains to the entire library. Appendix E contains more detailed information about the contents of this coverage.

5.2.2.4.6 Names placement coverage. The names placement coverage provides the user with a way to locate a place in a library by using a place name. This is a special type of thematic query. The most common use of the names placement coverage is to enter a query string (for instance "London"), have the software locate all the places that are "London," and display their geographic locations and names on the display device. The name feature class contains a point feature table and an entity node primitive table.

5.2.2.5 Database. A database is a collection of related libraries and additional tables. The library attribute table acts as a table of contents for the database. Database transmittal information is contained in a database header table. Database level data quality information can be maintained in the data quality table. Appendix E contains more detailed information about the content of this table. Additional metadata that is product specific may be stored in VPF tables in the product-specific directory, or additional columns may be added to the library attribute table and the database header table. Figure 17 illustrates the arrangement of database tables and coverages.

FIGURE 17. Database directory.

5.2.3 Data quality. VPF allows for the storage of data quality information to permit the evaluation of the data for particular applications. Although the exact form of the data quality information supplied for a database is set by a product specification, VPF supports incorporation of data quality information at each structural level in the database (see table 9). Data quality information may be stored at any VPF level. When it exists at a given level, it applies to all data at or below that level. However, when data quality information exists at multiple levels, the information stored at lower levels always takes precedence over that at the higher levels.

TABLE 9. VPF data quality information.

Level	Quality Attributes	Quality Coverages
Database	In the data quality table within the database directory.	Within the database directory.
Library	In the data quality table within the library directory.	Within the library directory.
Coverage	In the data quality table within the coverage directory.	A standalone data quality coverage.
Feature	In the feature attribute table.	Not applicable.
Primitive	In the primitive table.	Not applicable.

5.2.3.1 Types of data quality information. A VPF database may contain seven types of data quality information: source, positional accuracy, attribute accuracy, date status, logical consistency, feature completeness, and attribute completeness. Definitions of these quality types are provided in appendix E. The extent of data quality information contained in a product and the types of data quality to be included are determined by the product specification.

5.2.3.2 Data quality encoding. As shown by table 9, data quality information can be represented as an attribute or as a coverage. In the case of attributes, data quality information may be added to an existing VPF table, stored in a separate table, or stored in the data quality table discussed in section 5.3.7. The data quality coverage will maintain level 3 topology, where area or complex features designate faces with uniform data quality information of specified types. Appendix E describes data quality encoding in more detail.

5.3 Implementation. The following paragraphs describe the implementation requirements of the VPF data structures. Discussion covers the primitive, feature class, coverage, library, and database levels. A description of a data quality table and the narrative table is also provided.

5.3.1 General implementation information. In order to fully explain the content of each data structure, each table is given a text description, definition table, and an example.

5.3.1.1 Table definitions. These column descriptions define the contents of each table. Each description example contains five entries: column name, description, column data type, key type, and whether the column is optional or mandatory (Op/Man; see Table 13). The asterisk (*) in the column name item is a substitute for an associated feature or primitive table name that is provided by the product specification. Table 10 is an example of the table definition style.

TABLE 10. Table definition example.

Column Name	Description	Field Type	Key Type	Op/Man
ID	Entity node primary key	I	P	M
*.PFT_ID	Feature id reference	I	N	OF
CONTAINING_FACE	Face containing the entity node (can be used as foreign key to face primitive table)	I	N	M3
FIRST_EDGE	(Null)	X	N	O
COORDINATE	Entity point coordinates	C	N	M

Schema descriptions identify the following columns.

- a. Column type. The column type column in the definition table expresses the type of data the column must contain. The encapsulation of these types is discussed in additional detail in section 5.4. For the purposes of this section, table 11 identifies field types.
- b. Key type. VPF provides three key types. They are primary keys, unique keys, and non-unique keys. For purposes of this document, all columns that are not primary keys are considered to be non-unique, even if, in fact, each value in the relevant column is different. If a column in a specific product contains only unique values, the column can be specified in the product specification as being of the unique type. Table 12 lists the key types and the codes used in table definitions.

TABLE 11. Field types.

Column Type	Description
T,n	Fixed-length text
T,*	Variable-length text
F,1	Short floating point
R,1	Long floating point
S,1	Short integer
I,1	Long integer
C,n	2-coordinate array short floating point
C,*	2-coordinate string
B,n	2-coordinate array long floating point
B,*	2-coordinate string
Z,n	3-coordinate array short floating point
Z,*	3-coordinate string
Y,n	3-coordinate array long floating point
Y,*	3-coordinate string
D,1	Date and time
X,1	Null field
K,1	Triplet id

Note: The asterisk (*) indicates variable-length string. n indicates a fixed-length array; n is defined by the product specification. Type characters are case sensitive when used in table definitions.

TABLE 12. Key types.

Key	Description
P	Primary key
U	Unique key
N	Non-unique key

- c. Optional/mandatory. The optional/mandatory column indicates whether the column is optional or mandatory for a VPF table. For each column, there are several mandatory conditions, as shown in table 13. The code OF is used on a primitive table when direct pointers to the feature table are desired to improve performance. This is recommended when tiles exist in the coverage.

TABLE 13. Optional/mandatory conditions.

Column Name	Description
O	Optional
OF	Optional feature pointer
M	Mandatory
M<n>	Mandatory at level n topology (0-3)
MT	Mandatory if tiles exist

5.3.1.2 Reserved table names and extensions. Each VPF table name consists of a reserved name or suffix extension. Table 14 lists the tables whose names cannot be modified or changed.

There are a few reserved directory names at the library and database levels. These names are listed in table 15.

In a coverage directory, there are many feature class tables that have reserved suffixes. The product specification may define any eight-character prefix, following the naming conventions detailed in section 5.4.5. Table 16 lists the table suffixes.

TABLE 14. Reserved file names.

File Name	Description
CAT	Coverage Attribute Table
CND	Connected Node Primitive
CSI	Connected Node Spatial Index
DHT	Database Header Table
DQT	Data Quality Table
EBR	Edge Bounding Rectangle
EDG	Edge Primitive
END	Entity Node Primitive
ESI	Edge Spatial Index
FAC	Face Primitive
FBR	Face Bounding Rectangle
FCS	Feature Class Schema Table
FSI	Face Spatial Index
GRT	Geographic Reference Table
LAT	Library Attribute Table
LHT	Library Header Table
NSI	Entity Node Spatial Index
RNG	Ring Table
TXT	Text Primitive
TSI	Text Spatial Index
CHAR.VDT	Character Value Description Table
INT.VDT	Integer Value Description Table

TABLE 15. Reserved directory names.

Directory Name	Description
LIBREF	Library reference coverage
DATAQUAL	Data quality coverage
TILEREF	Tile reference coverage
PRODSPEC	Product-specific directory
GAZETTE	Names placement coverage

TABLE 16. Reserved table name extensions.

File Name Suffix	Description
.AFT	Area Feature Table
.AJT	Area Join Table
.ATI	Area Thematic Index
.CFT	Complex Feature Table
.CJT	Complex Join Table
.CTI	Complex Thematic Index
.DOC	Narrative Table
.LFT	Line Feature Table
.LJT	Line Join Table
.LTI	Line Thematic Index
.PFT	Point Feature Table
.PJT	Point Join Table
.PTI	Point Thematic Index
.TFT	Text Feature Table
.TTI	Text Thematic Index

Any table that contains variable-length records must have a variable-length index associated with it. The index file shall have the same file name as the table, except that the last character will end with "X." For example, a variable-length record road line table, ROAD.LFT, would have a variable-length index ROAD.LFX.

5.3.2 Primitives. As discussed in section 5.2.2.1, there are three types of geometric primitives in VPF: nodes, edges, and faces. There are two classes of nodes, the entity node and the connected node. In addition, text is used as a cartographic primitive. These four primitives, with the addition of feature tables, allow the modeling of geographic phenomena requiring vector geometry. Figure 18 illustrates the various types of primitives.

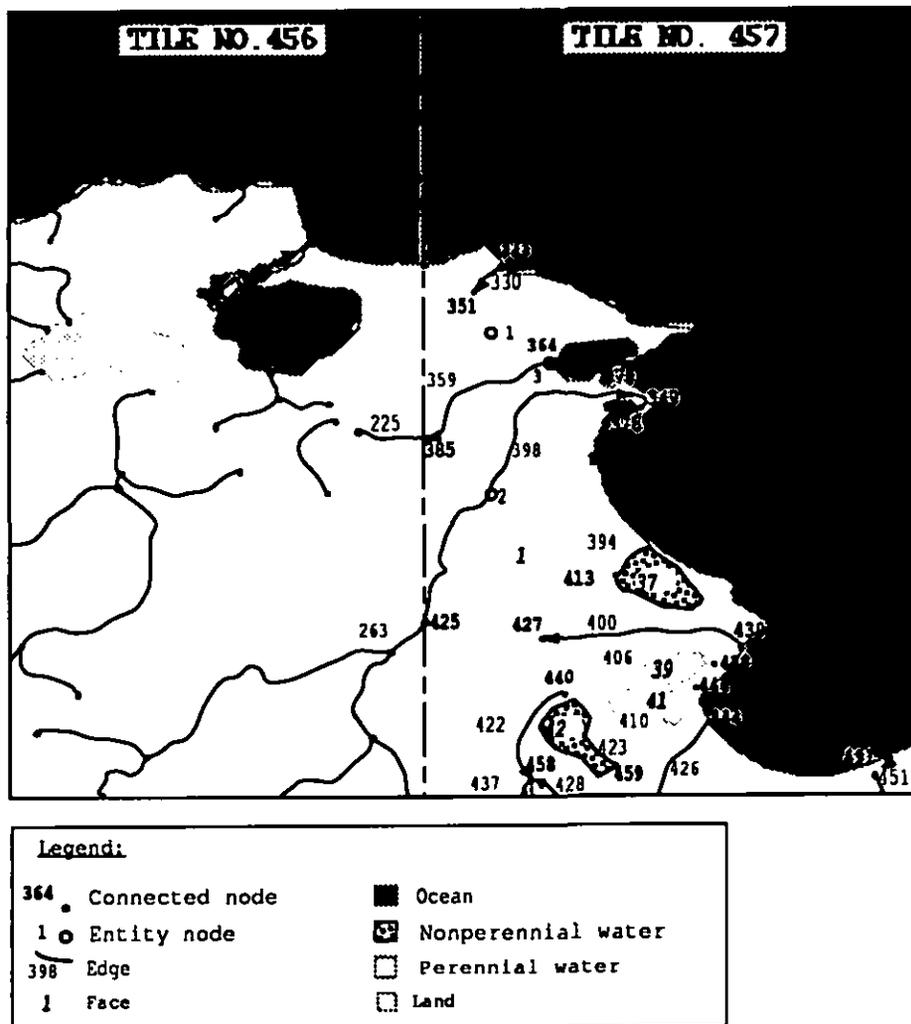


FIGURE 18. Node, edge, and face primitives.

5.3.2.1 Node primitives. Two types of node primitives are implemented: entity nodes, which are free floating, and connected nodes, which occur only at edge ends. Both represent zero-dimensional locations.

- a. Entity node primitive. The entity node primitive is composed of three columns: a primary key, a foreign (to the face table) key, and the node coordinates. The FIRST_EDGE null column is included to maintain compatibility with the connected node primitive so that the formats for both classes of node primitive conceptually remain the same. The CONTAINING_FACE column is only required for level 3 topology to maintain a topological relationship to the face that contains the node. Table 17 defines the meaning of the entity node primitive. Table 18 illustrates an entity node table; the entity nodes described are those in figure 18.

TABLE 17. Entity node definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Entity node primary key	I	P	M
*.PFT_ID	Feature id reference	I	N	OF
CONTAINING_FACE	Face containing the entity point (can be used as a foreign key to face in a primitive table)	I	N	M3
FIRST_EDGE	(Null)	X	N	O
COORDINATE	Entity point coordinates	C/Z/B/Y	N	M

Note: The asterisk (*) indicates a placeholder for the point feature class name.

TABLE 18. Entity node records.

Column Name	Description
ID	1
DNPOINT.PFT_ID	936
CONTAINING_FACE	Null
FIRST_EDGE	Null
COORDINATE	10.13, 37.15
ID	2
DNPOINT.PFT_ID	937
CONTAINING_FACE	Null
FIRST_EDGE	Null
COORDINATE	10.07, 37.02
ID	3
DNPOINT.PFT_ID	953
CONTAINING_FACE	1
FIRST_EDGE	Null
COORDINATE	10.07, 37.19

- b. Connected node primitive. The connected node primitive is composed of three columns: a primary key, a foreign key (to the edge table), and the node coordinates. The CONTAINING_FACE null column is included to maintain compatibility with the entity node primitive. The FIRST_EDGE column is a foreign key required for level 1 and higher topology levels to maintain a topological relationship to the edges that include the node. The complete set of edges around a connected node may be assembled by following the topology of the connected node until the first edge reappears. Refer to appendix B for more discussion of this algorithm. Table 19 defines the connected node primitive. Table 20 illustrates a connected node table with data for three of the connected nodes shown in figure 18.

TABLE 19. Connected node definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Node primary key	I	P	M
*.PFT_ID	Point feature table id	I	N	OF
CONTAINING_FACE	Always null (included for compatibility)	X	N	O
FIRST_EDGE	Edge key (foreign key to the edge table)	I	N	M1-3
COORDINATE	Node coordinates	C/Z/B/Y	N	M

Note: The asterisk (*) indicates a placeholder for the point feature class name.

TABLE 20. Connected node records.

Column Name	Description
ID	343
DN.PFT_ID	Null
CONTAINING_FACE	Null
FIRST_EDGE	330
COORDINATE	10.08, 37.25
ID	351
DN.PFT_ID	Null
CONTAINING_FACE	Null
FIRST_EDGE	330
COORDINATE	10.05, 37.22
ID	364
DN.PFT_ID	Null
CONTAINING_FACE	Null
FIRST_EDGE	359
COORDINATE	10.13, 37.15

5.3.2.2 Edge primitive. The edge primitive is implemented as follows. The edge primitive table will include eight columns: a row id (the primary key); start and end nodes (foreign keys to the connected node, if it exists); left and right face (foreign keys to the face table, if it exists); left and right edge (foreign keys within the same edge table); and the edge coordinate string. For simplicity in drawing edges, the coordinate string includes the node coordinates at each end, regardless of the existence of a connected node primitive. Thus, the minimum length of the coordinate string is two pairs. Appendix B describes winged-edge topology in more detail.

- a. Node information. The foreign keys to the connected node table are required for level 1 and higher topology levels to maintain a topological relationship to the node connected to the edge. The start node indicates the beginning of the edge, and the relationship of the start node to the end node defines the edge orientation.
- b. Edge information. Two foreign keys, right and left edge, are required for level 1, 2, and 3 topology, establishing connectivity between each edge and its neighboring edges in the coverage network. The right and left edges establish winged-edge topology for both line networks and faces. If all the edges incident at a node are sorted according to the bearing each edge radiates from that node, the right edge of a particular edge is the first edge encountered, counterclockwise in the sort order, around the end node of that particular edge. Similarly, the left edge is the first edge encountered around the start node.
- c. Face information. When faces are present, the right and left face columns are added to the edge primitive table. Depending on the edge direction, the face columns are assigned. When a face is split by a tile boundary, the internal tile boundary is used to close the face on each tile. The tile id and external face id are also maintained in the triplet id.

Table 21, defines the edge table. Table 22 illustrates an edge table with data for the 14 drainage edges of tile 457 in figure 18.

TABLE 21. Edge table definition.

Column Name	Description	Field Type	Key Type	Op/Man
ID	Edge primary key	I	P	M
**LFT_ID	Line feature table id	I	N	OF
START_NODE	Start node (foreign key to the node primitive)	I	N	M1-M3
END_NODE	End node (foreign key to the node primitive)	I	N	M1-M3
RIGHT_FACE	Right face (foreign key to the face primitive)	K/I	N	M3
LEFT_FACE	Left face (foreign key to the face primitive)	K/I	N	M3
RIGHT_EDGE	Right edge from end node (foreign key to edge primitive)	K/I	N	M1-M3
LEFT_EDGE	Left edge from start node (foreign key to edge primitive)	K/I	N	M1-M3
COORDINATES	Edge coordinates	C/Z/B/Y,*	N	M

Note: The (**) indicates a placeholder for the line feature class name.

TABLE 22. Edge record example.

ID	LINE	SN	EN	RF	LF	REd	LEd	Coordinates	
								Start Node...	End Node
330	24505	343	351	1	1	330	330	10.08, 37.25...	10.05, 37.22
349	24524	372	376	1	1	349	398	10.22, 37.11...	10.23, 37.10
359	24534	364	385	1	1	359	359	10.13, 37.15...	10.00, 37.07
394	24569	413	413	1	37	394	394	10.24, 36.95...	10.24, 36.95
398	24573	425	372	1	1	349	398	10.00, 36.88...	10.22, 37.11
400	24575	430	427	1	1	400	400	10.33, 36.85...	10.12, 36.86
406	24581	438	438	1	39	406	406	10.20, 36.81...	10.20, 36.81
410	24585	446	446	1	41	410	410	10.26, 36.77...	10.26, 36.77
422	24597	440	458	1	1	437	422	10.15, 36.80...	10.11, 36.72
423	24598	459	459	42	1	423	423	10.18, 36.72...	10.18, 36.72
426	24601	444	462	1	1	430	426	10.30, 36.78...	10.24, 36.69
428	24603	466	458	1	44	422	19	10.14, 36.64...	10.11, 36.72
437	24612	458	477	1	44	439	428	10.11, 36.72...	10.03, 36.55
451	24626	491	457	1	1	451	451	10.39, 36.50...	10.47, 36.72

Note: LINE = **LFT_ID, SN = START_NODE, EN = END_NODE, RF = RIGHT_FACE, LF = LEFT_FACE, REd = RIGHT_EDGE, LEd = LEFT_EDGE. Only start and end node coordinates are shown, although all coordinates would actually be present in this variable-length column.

5.3.2.3 Face primitive. Faces are defined as planar regions enclosed by an edge or a set of edges. All faces are defined by one or more rings, which are connected networks of edges that compose the face border. Each ring starts with a reference to a particular edge, and is defined by traveling in a consistent direction. Then the left and right edge columns on the edge primitive are traversed, always keeping the face being defined on one side, until the ring returns to its starting edge. All faces must have one and only one outer ring, which bounds the exterior. A face may require inner rings to represent areas belonging to other faces that it encloses totally. Inner rings and outer rings are disjointed. There is no upper limit on the number of inner rings. A ring table (see below) is defined to handle these disjointed rings. Face primitives are implemented as follows.

- a. Face table. The face table contains two columns, where the primary key id identifies the face and the RING_PTR column points to the outer ring in the ring table. Face id 1 is always reserved for the universe face in a face table. The universe face contains a point at infinity. The outer ring of the universe face is a topological artifact which does not have a geometric representation. The outer ring cannot be displayed. The common boundary between the universe face and all other faces constitutes the inner ring or rings of the universe face. Inner rings of the universe face behave the same as the rings of other faces. Table 23 defines the face table. Table 24 depicts an example of the face table for three faces from figure 18.

TABLE 23. Face table definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Face primary key	I	P	M3
*.AFT_ID	Area feature table id	I	N	OF
RING_PTR	Ring pointer (can be used as foreign key to ring table)	I	N	M3

Note: The asterisk (*) indicates a placeholder for the area feature class name.

TABLE 24. Face record example.

Column Name	Contents
ID	1
DNAREA.AFT_ID	4570
RING_PTR	1
ID	2
DNAREA.AFT_ID	4571
RING_PTR	37
ID	3
DNAREA.AFT_ID	4572
RING_PTR	38

- b. Ring table. The ring table contains one reference to the edge table for each ring of a face. The first row in the ring table for each face refers to the outer ring of that face. Outer rings are traversed in clockwise direction. Each inner ring has one reference to a first edge on that ring. The ring table maintains an order relationship for its rows. The first record of a new face id will always be defined as the outer ring. Any repeating records with an identical face value will define inner rings. Table 25 defines ring table structure, and table 26 depicts three rings from figure 18 and follows the face example in table 24.

TABLE 25. Ring table definition.

Column Name	Description	Field Type	Key Type	Op/Man
ID	Ring primary key	I	P	M3
FACE_ID	Face id (row id from face table)	I	N	OF
START_EDGE	First edge in the ring (foreign key to edge primitive)	I	N	M3

TABLE 26. Ring record example.

Column Name	Contents
ID	1
FACE_ID	1
START_EDGE	36
.	.
.	.
.	.
ID	18
FACE_ID	1
START_EDGE	394
.	.
.	.
.	.
ID	37
FACE_ID	2
START_EDGE	36
.	.
.	.
.	.

Note: Gaps in row sequence have been left to permit coding for rings from different faces to be illustrated.

5.3.2.4 Text primitive. Text is implemented to allow the representation of names associated with vague or ill-defined regions, such as the Appalachian Mountains. The text primitive is normally composed of three items: a primary key, the text string, and a coordinate string defining a shape line. Optional attributes may also be associated with the primitive, such as text color or font height.

The shape line must contain at least one coordinate pair. If the shape line contains only one coordinate pair, the coordinate pair is considered to represent the lower left coordinate, and the default orientation for the shape line will be assumed (minimum readable text and parallel to X axis.) In order to specify orientation, two coordinate pairs must be entered. The second coordinate pair defines the lower right of the string. Some fonts have ascenders and descenders that extend above or below the shape line. Third and subsequent coordinate pairs define control points in a shape line. The control points of a shape line define a continuous function. Characters in a text string are individually oriented along the shape line.

Table 27 defines the text primitive table structure. Table 28 is a hypothetical example of a text primitive table.

5.3.2.5 Minimum bounding rectangle table. A minimum bounding rectangle record is required for each record in an edge or face primitive table. The definition found in table 29 is used for both the face and edge minimum bounding rectangle tables. Table 30 is an example of the face minimum bounding rectangle table used for figure 18. The MBR table definition is applicable to both edge and face primitives.

TABLE 27. Text primitive structure table.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Text primary key	I	P	M
**TXT_ID	Text feature id	I	N	OF
STRING	The text string	T,*	N	M
SHAPE LINE	The shape line	C/Z/B/Y,*	N	M

Note: The (**) indicates a placeholder for the text feature class name.

TABLE 28. Text primitive record example.

Column Name	Contents
ID	1
DNTEXT.TFT_ID	529
STRING	Fiume Salso
SHAPE_LINE	14.41,37.74 14.45,37.73 14.52,37.69 14.60,37.69
ID	2
DNTEXT.TFT_ID	530
STRING	Simeto
SHAPE_LINE	14.80,37.71 14.81,37.68 14.80,37.66 14.81,37.65
ID	3
DNTEXT.TFT_ID	531
STRING	Belice
SHAPE_LINE	12.91,37.69 12.93,37.71 12.95,37.73
ID	4
DNTEXT.TFT_ID	532
STRING	Dittaino
SHAPE_LINE	14.51,37.56 14.54,37.55 14.58,37.56 14.62,37.57

TABLE 29. Minimum bounding rectangle definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Row id	I	P	M
XMIN	Minimum x coordinate	F/R	N	M
YMIN	Minimum y coordinate	F/R	N	M
XMAX	Maximum x coordinate	F/R	N	M
YMAX	Maximum y coordinate	F/R	N	M

TABLE 30. Face bounding rectangle record example.

Column Name	Contents
ID	1
XMIN	10.00
YMIN	35.00
XMAX	15.00
YMAX	38.19
ID	2
XMIN	12.31
YMIN	37.97
XMAX	13.31
YMAX	37.99
ID	3
XMIN	14.54
YMIN	37.83
XMAX	14.58
YMAX	37.85

5.3.3 Feature class. A feature class is composed of a variety of tables containing geometric, topologic, and attribute data. Complex feature relationships can be defined. Some features may require a single feature and an associated primitive table, while others may need multiple tables linking features into a complex hierarchy. The feature class definition is provided by a product specification.

Constructing feature classes requires the use of feature tables. If necessary, the feature class definition may require the use of a feature join table to accurately construct the feature in relational form. Appendix C describes feature class relationships in greater detail.

5.3.3.1 Feature tables. A feature table consists of a feature identifier and one or more attribute columns. Table 31 defines feature table contents. If the coverage is tiled, the feature table maintains a triplet id column that identifies the tile id and primitive id that are related to the feature. This column is named after the primitive table it relates to, followed by "_ID." Alternatively, to support some data processing environments that cannot interpret bit fields efficiently, the tile id and the primitive id can be stored in separate columns. The tile id column is named TILE_ID.

TABLE 31. Feature table definition.

Column Name	Description	Column Type	Key	Op/Man
ID	Feature primary key	I	P	M
TILE_ID	Tile reference ID	S	N	MT
*_ID	Primitive id	S/I/K	N	MT
<Attribute 1>	First attribute value	Any	Any	M
<Attribute n>	nth attribute	Any	Any	O

Note: The asterisk (*) indicates a placeholder for the primitive table name: EDG, FAC, END, CND, or TXT. If a join table exists, both the TILE_ID and *_ID columns will be found there.

All feature class definitions fall into one of five categories: area, line, point, text, and complex feature tables.

- a. Area feature. An area feature is composed of face primitives. The primitive id column name will be FAC_ID. For instance, a vegetation feature will be defined by face primitives and will have various descriptive attributes (such as canopy closure, stem diameter size, and vegetation type category).
- b. Line feature. Line features are composed of edge primitives. The primitive id column name will be EDG_ID.

A river feature could contain the following attributes: hydrographic category, depth, and width.

- c. Point feature. A point feature class contains node primitives. The primitive id column name will be END_ID or CND_ID. For instance, a dam feature class may contain many attributes, such as height, length, and width.
- d. Text feature. A text feature class is composed of a text primitive. The primitive id column name will be TXT_ID. The text feature usually contains additional attributes, such as text color, font, and font size.
- e. Complex feature. Complex features can be constructed from any features. Complex features, however, cannot be recursively defined.

5.3.3.2 Feature join tables. Feature join tables are used to combine multiple feature tables into the relational form. These join tables allow a variety of constructs to be made, usually with one or more feature tables and one or more primitive tables. A feature join table connects the rows of the feature table with the rows of a feature or primitive table. A join is used to relate one column in a table with a column in a second table. This is common when a 1:N relationship exists between two tables. Thus, for example, if a complex feature has four 1:N relations with four simple features, four join tables will be used.

Unlike feature tables that have four categories (area, line, point, and complex), the same feature join table is applicable to all feature classes. Table 32 defines feature join table contents. The second column contains the key of the first table in the join. The name of this column is the table's name followed

TABLE 32. Feature join table definition.

Column Name	Description	Column Type	Key	Op/Man
ID	Row id	I	P	M
*_ID	Feature key	I	N	M
TILE_ID	Tile reference ID	S	N	MT/O
**_ID	Feature or primitive primary key	I/S/K	N	M

Note: The (*) indicates a placeholder for the name of the first table in the join, usually a feature class table name. The (**) indicates a placeholder for the table name of the second table in the join, usually one of EDG, FAC, END, CND, or TXT.

by "_ID." The third column contains the key of the second table in the join; it is similarly labeled using the table name and the "_ID" suffix. For instance, in a feature table (SDRPOINT.PFT) relating to a primitive (END), the key for the first table is

SDRPOINT.PFT_ID and the key for the second table is END_ID. The feature class schema table is used to express the column names for the join table. The TILE_ID column is mandatory when tiled directories exist in the coverage and the join table relates a feature table and a primitive table. Refer to section 5.3.3.3 concerning these columns when tile directories exist in the coverage.

5.3.3.3 Feature-to-primitive relations on tiled coverages.

If the coverage is tiled, the feature or join table maintains a triplet id column that identifies the tile id and primitive id that are related to the feature. This column is named after the primitive table it relates to, followed by "_ID." The first field in the triplet id is null. The second field is the tile id (found in the tile reference coverage); the third field is the primitive row id in that tile. If the relation between the primitive and the feature is made using a join table, then this information will be stored in the join table unless it is stored on the feature table.

5.3.4 Coverage. Each coverage has one set of topological primitives and a collection of feature tables based upon these topological objects. All these tables are stored in one directory and are associated by file naming conventions (see table 16). The coverage may contain a data dictionary for all feature tables in the coverage. An optional data quality table is allowed at the coverage level.

5.3.4.1 Coverage relationships. The general description of a coverage is stored in the coverage attribute table of its library. The relationships between the tables in a coverage are described by the mandatory feature class schema table.

5.3.4.2 Feature class schema table. A feature class schema table defines the feature classes that are contained within the coverage. Each record in the table specifies a feature class name, the name of the two tables involved in the join, and the names of the columns used in the join. The feature class name must be repeated to specify all the relationships in the feature class schema table. The topological relationships need not be specified, since they are implied by table types.

If a key in the join is a compound key, the column names will be listed, separated by a backslash character (\). For example, a primary key composed of two columns would be specified as "NAME\TYPE." Table 33 defines feature class schema contents, and table 34 is an example of a feature class table.

TABLE 33. Feature class schema definition.

Column Name	Description	Column Type	Key	Op/Man
ID	Row id	I	P	M
FEATURE_CLASS	The name of the feature class	T,8	N	M
TABLE1	The first table in a relationship	T,12	N	M
TABLE1_KEY	Join column in the first table	T,*	N	M
TABLE2	The second table in a relationship	T,12	N	M
TABLE2_KEY	Join column in the second table	T,*	N	M

TABLE 34. Feature class schema example.

Column Name	Contents
ID FEATURE_CLASS TABLE1 TABLE1_KEY TABLE2 TABLE2_KEY	1 SDRPOINT SDRPOINT.PFT ID END ID
ID FEATURE_CLASS TABLE1 TABLE1_KEY TABLE2 TABLE2_KEY	2 SDRPOINT END ID SDRPOINT.PFT ID
ID FEATURE_CLASS TABLE1 TABLE1_KEY TABLE2 TABLE2_KEY	3 SDRAREA SDRAREA.AFT ID FAC ID
ID FEATURE_CLASS TABLE1 TABLE1_KEY TABLE2 TABLE2_KEY	4 SDRAREA FAC ID SDRAREA.AFT ID

5.3.4.3 Value description table. A value description table relates to associated feature class tables within a coverage. There exists an integer VDT and a character VDT. There will be no more than one of each per coverage. If a column in a feature table requires a VDT, then every distinct column value will have an entry in the VDT. Table 35 defines the format of a VDT, and table 36 provides an example.

TABLE 35. Value description table definition.

Column Name	Description	Column Type	Key	Op/Man
ID	Row id	I	P	M
TABLE	Name of the feature table	T,12	N	M
ATTRIBUTE	Column name	T,16	N	M
VALUE	Unique value of attribute	I/S/T,n	N	M
DESCRIPTION	Description of attribute	T,50	N	M

TABLE 36. Integer value description record example.

Column Name	Contents
ID	1
TABLE	SDRPOINT.PFT
ATTRIBUTE	MCP
VALUE	0
DESCRIPTION	UNKNOWN
ID	2
TABLE	SDRPOINT.PFT
ATTRIBUTE	MCP
VALUE	18
DESCRIPTION	CONCRETE
ID	3
TABLE	SDRPOINT.PFT
ATTRIBUTE	MCP
VALUE	23
DESCRIPTION	EARTHEN
ID	4
TABLE	SDRPOINT.PFT
ATTRIBUTE	HYC
VALUE	0
DESCRIPTION	UNKNOWN
ID	5
TABLE	SDRPOINT.PFT
ATTRIBUTE	HYC
VALUE	6
DESCRIPTION	NON-PERENNIAL
ID	6
TABLE	SDRPOINT.PFT
ATTRIBUTE	HYC
VALUE	10
DESCRIPTION	TIDAL

5.3.5 Library. The function of a VPF library is to organize collections of coverages that pertain to one geographic region. Each library must manage its own spatial extent. VPF uses a minimum bounding rectangle to define this geographic extent. VPF also supports a coverage, the library reference coverage, which describes the library region in a graphic manner. All reference coverages must be spatially registered and be in the same coordinate system.

Within each library, there are three mandatory tables. The library header table defines the contents of the library. The geographic reference table contains information pertaining to the geographic location of the library. A coverage attribute table provides a list of and descriptions for the coverages contained in the library.

If the library is tiled, there will be an untiled reference coverage. There will also be an untiled library reference coverage that gives a general overview of the information contained in the library. In addition, it is possible to include a data quality coverage. The format for the data quality and library reference coverage follow the same rules as any normal VPF coverage. See appendix E for information concerning recommended data quality coverage contents. Multiple records in these tables are used to describe multiple sources, updates and maintenance issues.

5.3.5.1 Library header table. The library header table contains information identifying the library, general information about the contents, extent, projection, the units used, security, source, and data quality. Table 37 describes the group of entities that compose the library header table.

5.3.5.2 Geographic reference table. This table (table 38) contains four subrecords that define the geographic parameters of the library. These tables are the geographic parameters, projections, registration points, and diagnostic points. (Refer to appendix G for valid values in datum codes, projection codes, and unit of measure codes.)

5.3.5.3 Coverage attribute table. Each library will contain a set of coverages. Each coverage will have a name and a path name to its associated directory. In the case of tiled coverages, the library does not store the explicit path name, but it does identify the logical existence of a coverage under that path under each tile directory. The topological level associated with each coverage determines the nature of geometric and topological information available on that coverage. Table 39 defines coverage attribute table entities. Table 40 is an example of a coverage attribute table.

TABLE 37. Library header table.

Column Name	Description of Contents	Column Type	Key Type	Op/Man
ID	Row id	I	P	M
PRODUCT_TYPE	Series designator of product type	T,12	N	M
LIBRARY_NAME	Name of library	T,12	N	M
DESCRIPTION	Text description of library	T,100	N	M
DATA_STRUCT_CODE	Highest level code for the library: 5 - Level 0 topology 6 - Level 1 topology 7 - Level 2 topology 8 - Level 3 topology	T	N	M
SCALE	Source scale of the library (i.e. 200) using the representative fraction denominator	I	N	M
SOURCE_SERIES	Series designator (e.g., 1501)	T,15	N	M
SOURCE_ID	Source id - number or name, that when used in conjunction with the series and edition, will identify a unique source	T,30	N	M
SOURCE_EDITION	Source edition number	T,20	N	M
SOURCE_NAME	Full name of source document	T,100	N	M
SOURCE_DATE	Significant date. A designed date that most accurately describes the basic date of the product for computation of the probable obsolescence date. It can be the compilation date, revision date, or other depending on the product and circumstances	D	N	M
SECURITY_CLASS	Security classification of the source: T - TOP SECRET S - SECRET C - CONFIDENTIAL R - RESTRICTED (or alternatively "FOR OFFICIAL USE ONLY"; administrative classification only) U - UNCLASSIFIED	T	N	M
DOWNGRADING	Originator's permission for downgrading required (yes or no)	T,3	N	M
DOWNGRADING_DATE	Date of downgrading (null if answer to previous entity is yes)	D	N	M
RELEASABILITY	Releasability restrictions	T,20	N	M

TABLE 38. Geographic reference table.

Column Name	Description of Contents	Field Type	Key Type	Op/Man
ID	Row id	I	P	M
DATA_TYPE	Type of data in the library (GEO)	T,3	N	M
UNITS	Units of measure code for library	T,3	N	M
ELLIPSOID_NAME	Name of ellipsoid of the library	T,15	N	M
ELLIPSOID_DETAIL	Details about library ellipsoid including ellipsoid code	T,50	N	M
VERT_DATUM_NAME	Name of vertical reference	T,15	N	M
VERT_DATUM_CODE	Code of vertical datum reference	T,3	N	M
SOUND_DATUM_NAME	Name of sounding datum	T,15	N	M
SOUND_DATUM_CODE	Code of sounding datum reference	T,3	N	M
GEO_DATUM_NAME	Name of geodetic datum	T,15	N	M
GEO_DATUM_CODE	Code of geodetic datum	T,10	N	M
PROJECTION_NAME	Name of the projection	T,20	N	M
PROJECTION_CODE	Code of the projection, if any; null if geographic coordinates	T,2	N	O
PARAMETER1	Projection parameter 1	T,20	N	O
PARAMETER2	Projection parameter 2	T,20	N	O
PARAMETER3	Projection parameter 3	T,20	N	O
PARAMETER4	Projection parameter 4	T,20	N	O
FALSE_ORIGIN_X	False Easting origin of projection	T,20	N	O
FALSE_ORIGIN_Y	False Northing origin of projection	T,20	N	O
FALSE_ORIGIN_Z	False origin for Z values	T,20	N	O
REG_PT	Registration point id	T,10	N	O
REG_LONG	Longitude of registration point	T,15	N	O
REG_LAT	Latitude of registration point	T,15	N	O
REG_Z	Elevation of registration point	T,5	N	O
REG_TABLE_X	X table coordinate of control pts	T,15	N	O
REG_TABLE_Y	Y table coordinate of control pts	T,15	N	O
REG_TABLE_Z	Z table coordinate of control pts	T,15	N	O
DIAG_PT	Diagnostic point id	T,10	N	O
DIAG_LONG	Longitude of diagnostic point	T,15	N	O
DIAG_LAT	Latitude of diagnostic point	T,15	N	O
DIAG_Z	Elevation of diagnostic point	T,5	N	O
DIAG_TABLE_X	X table coordinate of diagnostic pts	T,15	N	O
DIAG_TABLE_Y	Y table coordinate of diagnostic pts	T,15	N	O
DIAG_TABLE_Z	Z table coordinate of diagnostic pts	T,15	N	O

TABLE 39. Coverage attribute table definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Row id	I	U	M
COVERAGE_NAME	The coverage name	T,8	P	M
DESCRIPTION	Description string	T,50	N	M
LEVEL	The topologic level	I	N	M

TABLE 40. Coverage attribute table example.

Column Name	Contents
ID COVERAGE_NAME DESCRIPTION LEVEL	1 SLP Surface configuration 3
ID COVERAGE_NAME DESCRIPTION LEVEL	2 VEG Vegetation 3
ID COVERAGE_NAME DESCRIPTION LEVEL	3 SMC Surface materials 3
ID COVERAGE_NAME DESCRIPTION LEVEL	4 SDR Surface drainage 3
ID COVERAGE_NAME DESCRIPTION LEVEL	5 TRN Transportation 3
ID COVERAGE_NAME DESCRIPTION LEVEL	6 OBS Obstacles 3

5.3.5.4 Tile reference coverage. A library may contain tiled partitions and will require a tile reference coverage, TILEREF, with associated area feature and attribute tables. Thus, if the tile reference coverage does not exist in a library, then no tiling scheme exists. The tile reference coverage consists of a set of faces identifying the tiles that the library uses to subdivide the region of interest. Tile attributes are used to optimize retrievals. At the library level, the tile geometry is simple, describing the location of the tile boundaries.

5.3.5.5 Tile attributes. The tile attributes are located in the area feature table (AFT) of the tile reference coverage. This

table maintains a unique id and its associated tile directory name. This directory name may contain a nested directory list or path name, relative to the coverage level. If a path name is required, the separator character will be a backslash ('\'). Additional columns may exist for every feature class or coverage located in the library, but this is optional. The id of the area feature table will be used in the tile and primitive id for each coverage feature table (see section 5.3.3.1).

Table 41 defines the tile reference coverage attribute columns. Table 42 is an example of a tile reference coverage area feature table.

TABLE 41. Tile reference area feature table definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Tile primary key	I	P	M
TILE_NAME	Tile directory name	T30	N	M

TABLE 42. Tile area feature record example.

Column Name	Contents
ID TILE_NAME	1 M\J\12
ID TILE_NAME	2 M\J\22
ID TILE_NAME	3 M\J\23
ID TILE_NAME	4 M\J\24

5.3.6 Database. Information that applies to the whole collection of data belongs at the database level. Structurally, a database consists of a set of libraries. It is possible to include a data quality coverage. The format for the data quality coverage follows the same rules as any normal VPF coverage. See appendix E for information concerning recommended data quality coverage content.

There are two mandatory tables: the library attribute table and the database header table. These two tables are described below. Multiple records in each table are used to describe multiple sources, updates, and maintenance issues.

5.3.6.1 Library attribute table. The library attribute table contains the name and extent for each library in the database. The library minimum bounding rectangle shall be in latitude and longitude (decimal degrees). Table 43 defines library attribute entities. Table 44 is an example of a library attribute table.

TABLE 43. Library attribute table entity definitions.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Row id	I	U	M
LIBRARY_NAME	Library name	T,8	P	M
XMIN	Westernmost Longitude	F/R	N	M
YMIN	Southernmost Latitude	F/R	N	M
XMAX	Easternmost Longitude	F/R	N	M
YMAX	Northernmost Latitude	F/R	N	M

TABLE 44. Library attribute table example.

Column Name	Contents
ID	1
LIBRARY_NAME	ITD
XMIN	-97.750000
YMIN	31.167000
XMAX	-97.667000
YMAX	31.250000

5.3.6.2 Database header table. The database header table (table 45) contains information that defines database content and security information.

5.3.7 Data quality. The data quality table may be stored at the database, library, or coverage level. It contains information on the completeness, consistency, date status, attribute accuracy, positional accuracy, and other miscellaneous quality information. It also contains information about the source from which the geographic data was derived. Lineage information should be included in the associated narrative table, named "LINEAGE.DOC." While the contents and location of a data quality table within a VPF database are product specific, it is highly recommended that at least one table exist at the library level. Table 46 defines the contents of the data quality table. Appendix E contains more information about storing data quality information in VPF.

TABLE 45. Database header table definition.

Column Name	Description of Contents	Column Type	Key Type	Op/Man
ID	Row id	I	P	M
VPF_VERSION	VPF version number	T,10	N	M
DATABASE_NAME	Directory name of the database	T,8	N	M
DATABASE_DESC	Text description of the database	T,100	N	M
MEDIA_STANDARD	Media standard used for the database	T,20	N	M
ORIGINATOR	Text for title and address of originator (a backslash "\" is used as a line separator)	T,50	N	M
ADDRESSEE	Text for title and address of addressee (a backslash "\" is used as a line separator)	T,100	N	M
MEDIA_VOLUMES	Number of media volumes comprising the database	T	N	M
SEQ_NUMBERS	Sequential number(s) for each media volume in this database	T	N	M
NUM_DATA_SETS	Number of libraries within database	T	N	M
SECURITY_CLASS	Security classification of database (the highest security classification of the transmittal including all datasets within the database) T - TOP SECRET S - SECRET C - CONFIDENTIAL R - RESTRICTED (or alternatively "FOR OFFICIAL USE ONLY") U - UNCLASSIFIED	T	N	M
DOWNGRADING	Originator's permission for downgrading required (yes or no)	T,3	N	M
DOWNGRADE_DATE	Date of downgrading	D	N	M
RELEASABILITY	Releasability restrictions	T,20	N	M
OTHER_STD_NAME	Free text, note of other standards compatible with this database	T,50	N	O
OTHER_STD_DATE	Publication date of other standard	D	N	O
OTHER_STD_VER	Other standard amendment number	T,10	N	O
TRANSMITTAL_ID	Unique id for this database	T	N	M
EDITION_NUMBER	Edition number for this database	T,10	N	M
EDITION_DATE	Creation date of this database	D	N	M

TABLE 46. Data quality table definition.

Column Name	Description of Contents	Column Type	Key Type	Op/Man
ID	Row id	I	P	M
VPF_LEVEL	Either DATABASE, LIBRARY, COVERAGE	T,8	N	M
VPF_LEVEL_NAME	Directory name of database, library, or coverage	T,8	N	M
FEATURE_COMPLETE	Feature completeness percent	T,*	N	M
ATTRIB_COMPLETE	Attribute completeness percent	T,*	N	M
LOGICAL_CONSIST	Logical consistency	T,*	N	M
EDITION_NUM	Edition number	T,8	N	M
CREATION_DATE	Date of creation	D	N	M
REVISION_DATE	Date of revision	D	N	M
SPEC_NAME	Name of product specification (e.g., DCW)	T,20	N	M
SPEC_DATE	Date of product specification	D	N	M
EARLIEST_SOURCE	Date of earliest source	D	N	M
LATEST_SOURCE	Date of latest source	D	N	M
QUANT_ATT_ACC	Standard deviation of quantitative attributes	T,*	N	O
QUAL_ATT_ACC	Percent reliability of qualitative attributes	T,*	N	O
COLLECTION_SPEC	Name of collection specification	T,*	N	M
SOURCE_FILE_NAME	Name of included source file	T,12	N	O
ABS_HORIZ_ACC	Absolute horizontal accuracy of database, library, or coverage	T,*	N	M
ABS_HORIZ_UNITS	Unit of measure for absolute horizontal accuracy	T,20	N	M
ABS_VERT_ACC	Absolute vertical accuracy of database, library, or coverage	T,*	N	M
ABS_VERT_UNITS	Unit of measure for absolute vertical accuracy	T,20	N	M
REL_HORIZ_ACC	Point to point horizontal accuracy of database, library, or coverage	T,*	N	M
REL_HORIZ_UNITS	Unit of measure for point to point horizontal accuracy	T,20	N	M
REL_VERT_ACC	Point to point vertical accuracy of database, library, or coverage	T,*	N	M
REL_VERT_UNITS	Unit of measure for point to point vertical accuracy	T,20	N	M
COMMENTS	Miscellaneous comments - free text	T,*	N	M

Multiple records in each table are used to describe multiple sources, updates, and maintenance issues.

5.3.8 Narrative table. The narrative table (table 47) will contain any descriptive information concerning its associated table. The contents of the TEXT column will be product specific.

TABLE 47. Narrative table definition.

Column Name	Description	Column Type	Key Type	Op/Man
ID	Row id	I	P	M
TEXT	Text information	T,80	N	M

5.3.9 Names placement coverage. The names placement coverage is optional at the library or database level. The names placement coverage must maintain the same bounding rectangle as the other library coverages (tile and library reference coverages). Other feature classes may exist, but are not required.

The names placement coverage will contain at least the following: a point feature table; an entity node primitive table; and a thematic index created on a fixed-length text column in the point feature table. The column's name will be "PLACE_NAME."

5.4 VPF encapsulation. Encapsulation defines the structure of data fields and the grouping of these fields. A simple table-oriented data encapsulation system is defined for VPF which allows the use of binary coded numeric data as well as text data and which uses references to identify the location of data elements within numerous related tables stored in files.

5.4.1 Table definition. A VPF table consists of a header and at least one record. Records can be of variable length; indexes can be used to directly access the files as needed. A table will begin with a header defining the table contents, followed by a series of records (rows). Where table records are not of fixed length, an external index containing record offsets and length (in bytes) will be used to provide direct access (see figure 19).

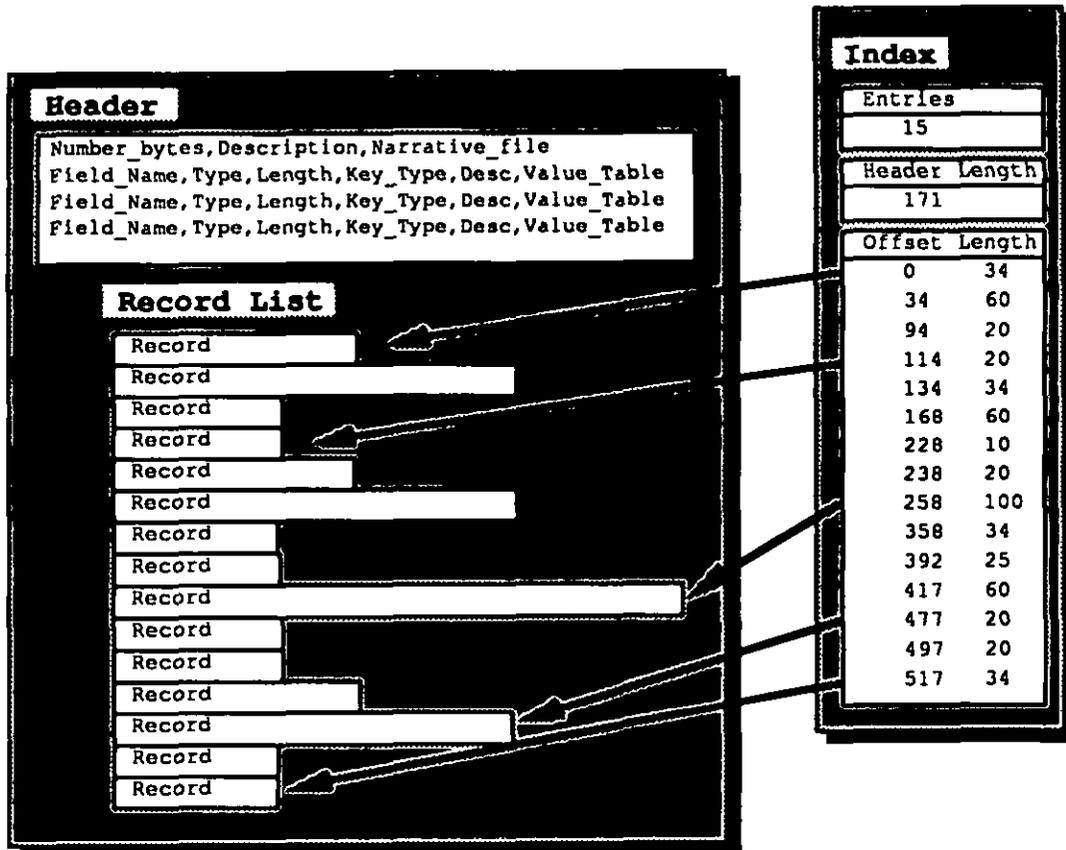


FIGURE 19. Table structure.

5.4.1.1 Header. A table's header (described in table 48) is composed of two parts. The first part is a 4-byte integer that indicates the length of the following text string, which defines the table. To accommodate different hardware architecture, after the length field a byte order field may be inserted. A value of "L" in the byte order field indicates a least-significant-byte-first encoding; an "M" indicates a most-significant-byte-first encoding. Least-significant-byte-first encoding is assumed if the byte order field is not present.

The second part, the header definition text string, contains three components, each of which is separated by a semicolon. The semicolon (;) indicates the end of the component.

TABLE 48. Header field definitions.

Field	Description	Field Type	No. Bytes
Header length	Length of ASCII header string (i.e., the remaining information)	I	4
Byte order	Byte order in which table is written: L - least-significant-first M - most-significant-first (semicolon separator)	T	1
		T	1
Table description	Text description of the table's contents (semicolon separator)	T,80	Up to 80
		T	1
Narrative table	An optional narrative file which contains miscellaneous information about the table (semicolon separator)	T,12	Up to 12
		T	1
Column definitions	The following fields repeat for each column contained in the table:		
Name	Name of the column (equal sign separator)	T,16	Up to 16
		T	1
Data type	One of the field types found in table 56 (comma separator)	T	1
		T	1
Number	Number of elements (comma separator)	T	1
		T	1
Key type	Key type (comma separator)	T	1
		T	1
Column description	Text description of the column's meaning (comma separator)	T,80	Up to 80
		T	1
Value description table name	Name of an associated value description table (comma separator)	T,12	Up to 12
		T	1
Thematic index	Name of thematic indexes. (comma separator)	T,12	12
		T	1
End of column	(colon separator) ... (repeat for each column)	T	1
End of header	(semicolon separator)	T	1

The first component is the table description. It contains a short table description and a file reference to a narrative text, if available. If no narrative file exists, the dash symbol (-) is used for the file reference. The final component is the column definition substring. The column definitions are separated by colons (:), which indicate the end of the subcolumn definition. If an entry is not applicable to the field (i.e., a thematic index does not exist), the dash symbol (-) is used to indicate a null value. Trailing null field entries need not be included. For

clarity in documentation, these trailing null fields should be listed, however. For each column in the table, there will be:

- a. Column name, followed by an equal sign (=)
- b. Data type indicator, followed by a comma (,)
- c. Number of data type elements, followed by a comma
- d. Key type indicator, followed by a comma
- e. Column description, followed by a comma
- f. Value description table name, if any, followed by a comma
- g. Thematic index name, if any, followed by a comma
- h. Column narrative file name, if any, followed by a comma

Table 49 displays an example of a text header. For presentation purposes, each component in the table definition string is listed on a separate line in table 49. No new line, space, or tab character should be inserted after the field and component separators in the table definition string.

Table 49 shows a header definition string for a Surface Drainage Area area feature table. A narrative file, SDR.DOC, is attached to this table in the VPF database. The table has 13 columns, with the column ID being the primary key column.

TABLE 49. Text header example.

```
Surface Drainage Area;
SDR.DOC;
ID=I,1,P,Required Row ID,-,:
F_CODE=T,5,N,FACS Code,CHAR.VDT,:
RGC=I,1,N,Railroad Gauge Category,INT.VDT,:
HYC=I,1,N,Hydrographic Category,INT.VDT,:
HFC=I,1,N,Hydrographic Form Category,INT.VDT,:
EXS=I,1,N,Existence Category,INT.VDT,:
WID=F,1,N,Width (meters) ,-, :
WV1=I,1,N,Water Velocity Average (m/sec),INT.VDT,:
WDA=I,1,N,Water Depth Average (meters),INT.VDT,:
MCP=I,1,N,Material Composition Primary,INT.VDT,:
DVR=I,1,N,Dense Bank Vegetation Right,INT.VDT,:
DVL=I,1,N,Dense Bank Vegetation Left,INT.VDT,:
BGR=I,1,N,Bank Gradient (Slope) Category Right Bank (%),,;
```

5.4.1.2 Record list. The body of data contained within the table is the record list; the header and the table index serve only to define the contents and provide effective access to this list. These records can be of fixed or variable length, as needed.

5.4.1.3 Variable-length index file. The variable-length index is a separate file that is mandatory when a VPF table contains variable-length records. As shown in table 50, the file

has two parts: a header and a data array. Each entry in the data array relates to a record in the VPF table.

TABLE 50. Components of variable-length index file.

File Component	Content of Component	Data Type	Field Length
Header	Number of entries (N) in index (which also matches the number of records in the associated table)	Integer	4 bytes
	Number of bytes in VPF table header	Integer	4 bytes
Data array	A two-dimensional array of N integers	Integer	4 bytes of N entities

The data array identifies the location of every record in the variable-length file by containing the following entries for each record:

[n][0] = Byte offset from beginning of file
 [n][1] = Number of bytes in table record

where n is an integer from 1 to N. The term byte offset refers to a location with respect to the beginning of a file. The first byte of a file has an offset of zero.

Thus, if the software requires the location of record 45 in a VPF table, the index file can be used to locate the exact position of the record without sequentially searching for the match. The entry for record 45 in the variable-length index would indicate the byte offset in the VPF table to the position of record 45 and the number of bytes in record 45.

5.4.2 Spatial index files. For each primitive (face, edge, entity node, connected node, and text), there can exist a spatial index file that will accelerate queries by software. Although these files are optional, they are recommended, especially for large libraries. These indexes are indirectly created on the coordinate column in each primitive; appendix F contains more information.

The format of the spatial index is as follows:

- a. Header record. The header will contain one integer defining the number of primitives (NUMPRIM) and another integer defining the number of nodes (NNODE) in the index. Following are four (xmin, ymin, xmax, ymax) long floating point coordinates defining the minimum bounding

rectangle. Table 51 shows the layout for the spatial index file header record.

TABLE 51. Spatial index file header record layout.

Byte Offset	Width	Type	Description
0	4	Integer	Number of primitives
4	4	Floating point	MBR x1
8	4	Floating point	MBR y1
12	4	Floating point	MBR x2
16	4	Floating point	MBR y2
20	4	Integer	Number of nodes in tree

- b. Bin array record. This record is a two-dimensional array the length of NUMPRIM described in the header record. The structure of this record is shown in table 52. This array maintains a long integer id and a long integer primitive count at that id.

TABLE 52. Structure of the bin array record.

Byte Offset	Width	Type	Description
HDR + n * 8	4	Integer	Offset of primitive list for node n
HDR + n * 8 + 4	4	Integer	Count in integer units

Note: n is {0 ... number of nodes-1}. HDR is the length of the index file header record.

- c. Bin data record. There are NUMPRIM records where each record contains an 8-bit bounding rectangle and the primitive id that falls within this rectangle. These primitive ids point into the associated primitive table. Table 53 shows the structure of the bin data record.

TABLE 53. Structure of the bin data record.

Byte Offset	Width	Type	Description
HDR+BIN + c * 8 + 0	1	byte	MBR x1
HDR+BIN + c * 8 + 1	1	byte	MBR y1
HDR+BIN + c * 8 + 2	1	byte	MBR x2
HDR+BIN + c * 8 + 3	1	byte	MBR y2
HDR+BIN + c * 8 + 4	4	int	Primitive id

Note: c is {0 ... number of primitives for a node-1}. HDR is the length of the index file header record. BIN is the length of the bin array record.

5.4.3 Thematic index files. A thematic index may be created for any column in an attribute table. There are two types of indexes, depending on the data content in a column: an inverted list thematic index or a bit array thematic index.

For categorical data or data with few distinct values, such as soil polygons where numerous polygons are assigned soil class designations from a relatively small number of classes, an inverted list is used. One entry in the index file is created for each distinct value in the column; correspondingly, a list of table record ids is stored with the value.

If the data in a column is all unique, especially in the case of an index for character strings, a bit array can be stored for each unique byte/character in the column. Each bit in the bit array represents a row in the indexed table. An 'ON' bit at a particular position means that the corresponding row in the table contains a specific byte/character pattern.

The character string form of the thematic index is used for names placement index implementations.

The thematic index file may be partitioned into three data groups: a fixed-length header, a variable number of index (or directory) entries (another index within an index), and a set of rows. Each row contains VPF record ids stored either as a list or as a bit array. Each directory entry describes the element being indexed and the location of the row containing the list (or set) of record ids related to the element.

- a. Header. The thematic index header contains 60 bytes of information that pertain to the type of index it is, the table it is associated with, and the column in that table. The layout of the header record is shown in table 54.
- b. Index directory. The index directory contains repeating records for each distinct element being indexed. The structure of an index directory record is shown in table 55. The number of entries is stored in the header record.

TABLE 54. Thematic index file header record layout.

Byte Offset	Width	Type	Description
0	4	Integer	60 bytes is the size of the header.
4	4	Integer	Number of directory entries. This is the number of items being indexed by a particular index file.
8	4	Integer	Number of rows in the data table from which this index file was derived.
12	1	Character	Type of index file; either "I" or "T" for inverted list index, or "B" or "G" for bit array index.
13	1	Character	Type of the data element being indexed; one of: "I"-4-byte integer "T"-character string "S"-2-byte integer "F"-4-byte floating point "R"-8-byte floating point
14	4	Integer	Number of data elements comprising one directory entry. This field will usually have a value of 1; an exception is a thematic index built on a text field.
18	1	Character	Type specifier for the data portion of an index file. Record ids in inverted list index can be stored by using either a 2-byte integer (type "S") or a 4-byte one (type "I"). Bit array index files always use type "S."
19	12	Character	The name of the VPF table from which the index file has been derived; no path information is included.
31	25	Character	The name of the column in the VPF table from which index entries have been pulled.
56	4	Character	Unused.

TABLE 55. Structure of index directory record.

Byte Offset	Width	Type	Description
HDR +n*(d+8)	d	Character Short floating point Integer Floating point Double precision	The element being indexed.
HDR +n*(d+8)+d	4	Integer	The offset from the beginning of the file to the location of the row associated with this index entry.
HDR +n*(d+8)+d+4	4	Integer	The number of indexed records associated with this entry. (For bit array index files, this is the number of bytes.)

Note: n is {0 ... number of index entries-1}, and d = size of (indexed type). HDR is the length of the index file header record.

- c. Index data. For each index entry there exists a data record consisting of either a list of row ids from the indexed file or a bit array.

5.4.4 Allowable field types. The field types depicted in table 56 are allowed and provide the ability to encode any data set. All variable-length types include a count item "n" (as depicted in Table 56) preceding the data. The count is a 4-byte integer. This count item contains the number of bytes in text strings and the number of tuples in coordinate strings.

5.4.5 Naming conventions. The following define the naming conventions for VPF file and column names. (See also the VPF reserved names in tables 14, 15, and 16.)

- a. All naming will use ISO 646 (ASCII) characters.
- b. For file names, the first character must be an alpha character from A to Z. The remaining 7 characters can be alphanumeric, including the underscore ('_') character.

A file name may have a trailing period with a 3-character suffix. The suffix may contain only characters from A to Z, except that X is not allowed.

Any table with variable-length records will maintain a variable-length index file with the same file name as its associated table except that the last character will be X.

TABLE 56. Allowable field types.

Abbrv	Column Type	Null/No Value	Length (bytes)
T,n	Fixed-length text	Blank filled	n
T,*	Variable-length text	Zero length	n + 4
F,1	Short floating point	NaN (not a number)	4
R,1	Long floating point	NaN	8
S,1	Short integer	-MAXSHORT	2
I,1	Long integer	-MAXLONG	4
C,n	2-coordinate array, short floating point	Both coordinates equal to null	8n
C,*	2-coordinate string	Length = 0	8n + 4
B,n	2-coordinate array, long floating point	Both coordinates equal to null	16n
B,*	2-coordinate string	Length = 0	16n + 4
Z,n	3-coordinate array, short floating point	All three coordinates equal to null	12n
Z,*	3-coordinate string	Length = 0	12n + 4
Y,n	3-coordinate array, long floating point	All three coordinates equal to null	24n
Y,*	3-coordinate string	Length = 0	24n + 4
D,1	Date and time	Space character filled	20
X,1	Null field	-	-
K,1	Triplet id	Type byte = 0	1-13

- c. All names are case insensitive, where they appear in uppercase.
- d. Directory names (names for libraries, databases, and coverages) are restricted to the same rules as for file names, except that there will be no suffix.
- e. Column names follow the same restrictions as file names, but they can be 16 characters in length. The dollar sign (\$), pound sign (#), dash ('-'), period ('.'), and slash ('/') are allowable characters.

The column name ("ID") is reserved and must be used to identify each table record.

- f. If a column is defined with a triplet id, the fields within the triplet id will be named as:

Field one	ID	The internal tile primitive id
Field two	TILE_ID	The tile reference id
Field three	EXT_ID	The external tile primitive id

The (\) will be used as a separator between the column name and the triplet id field. Thus, when referring to a the internal primitive id within a triplet id column (LEFT_FACE) the column name will be named "LEFT_FACE\ID."

5.4.6 Triplet id field type. As discussed in cross-tile keys (section 5.2.2.3.4), a triplet id can be used to reference primitives from multiple tiles in a tiled coverage. This field type replaces the integer foreign key used in untiled coverages. The triplet is composed of an 8-bit type byte. The type byte is broken down into four 2-bit pieces; each of these 2-bit pieces describes the length of a succeeding field. Table 57 lists the possible values for these 2-bit field descriptors. Only the first three fields are currently being used. The fourth field is reserved. Figure 20 is an example of the triplet id field.

TABLE 57. Type byte definitions.

Bit Count	Number Bits in Field
0	0
1	8
2	16
3	32

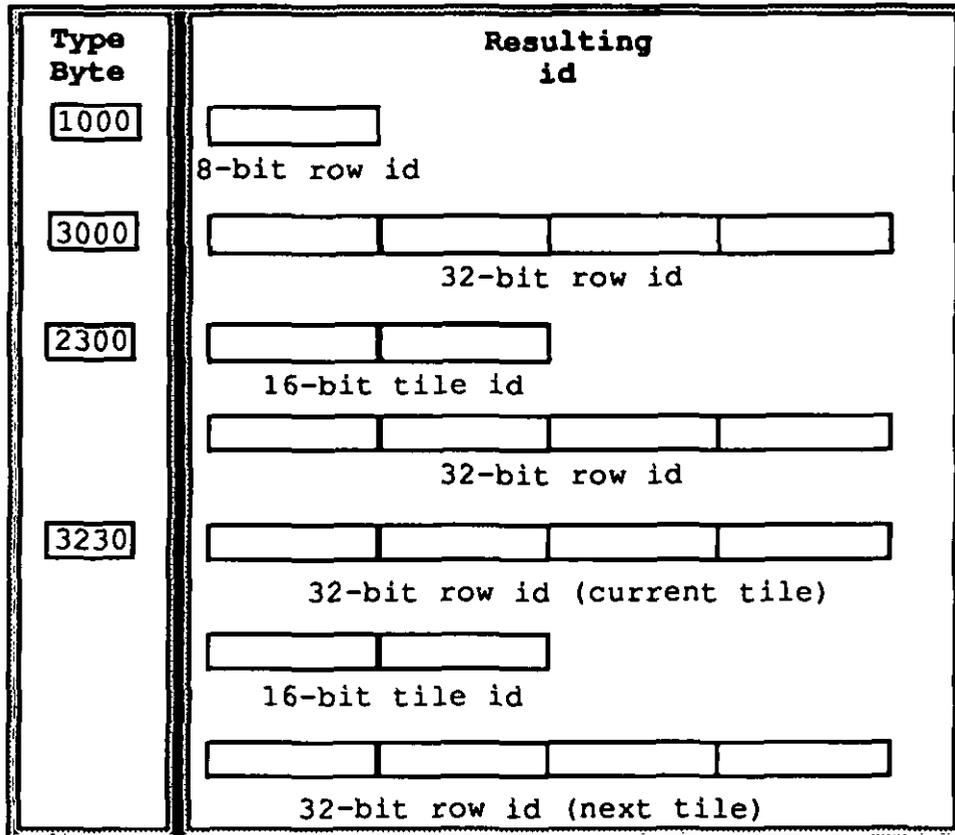


FIGURE 20. Examples of the triplet id.

The first field (referred to as ID) generally is used to store a primitive id without a tile id predicate. The tile reference id, the second field (TILE_ID), and the external primitive id, the third field (EXT_ID), together store an augmented primitive id for cross-tile topology.

5.5 Data syntax requirements. VPF requires the use of numeric, textual, coordinate, and date syntax items. These items comprise the lowest level of the VPF design. In order to utilize these items, a number of basic data types are required. These are integer or real numbers, strings of text, and coordinate data types. The coding of these data types is defined in terms of a number of international standards. VPF products may have a byte order specified in the product specification and the table header. Five categories of data syntax items are required in VPF. These are integer numbers, real numbers, text strings, coordinates, and date.

5.5.1 Integer numbers. The two fixed-length integer data fields are 16 bits and 32 bits; hereafter they are termed "short" and "long" precision, respectively. Integers are binary encoded using the 2's complement scheme. For integer number formats, the null value consists of the sign bit set to one and all trailing bits set to zero (-MAXINT). Different length integer numbers may be required in different situations. For example, the number 32,000 can be handled in 2 bytes of data, whereas the number 2,147,483,647 will fill 4 bytes of data. The general structure and several examples of the integer number format can be found in figure 21.

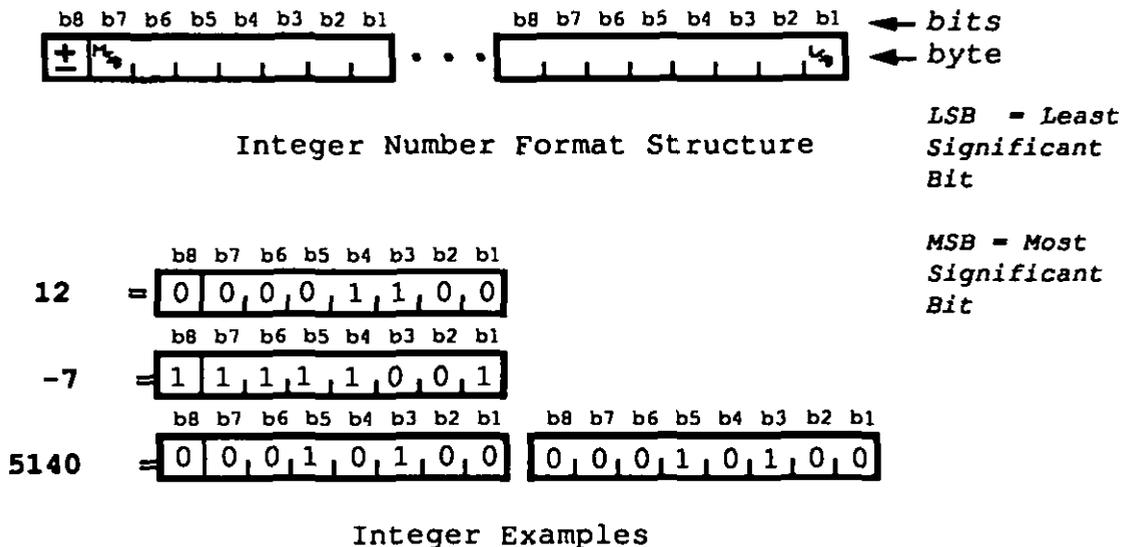


FIGURE 21. Integer number syntax.

5.5.2 Real numbers. Real numbers are needed to carry parametric information. VPF uses the IEEE floating-point real number format (ANSI/IEEE 754) in both 32-bit (short) and 64-bit (long) form. Numbers in the single and double formats are composed of the following three fields:

- s* 1-bit field for sign
- e* Biased exponent field (equals exponent *E* plus bias)
- f* Fraction field (mantissa)

a. Range. The range of the unbiased exponent includes every integer value between E_{min} and E_{max} , inclusive, and also two other reserved values, $E_{min} - 1$ and $E_{max} + 1$, to encode certain special states as described below. Figure 22 illustrates real number syntax.

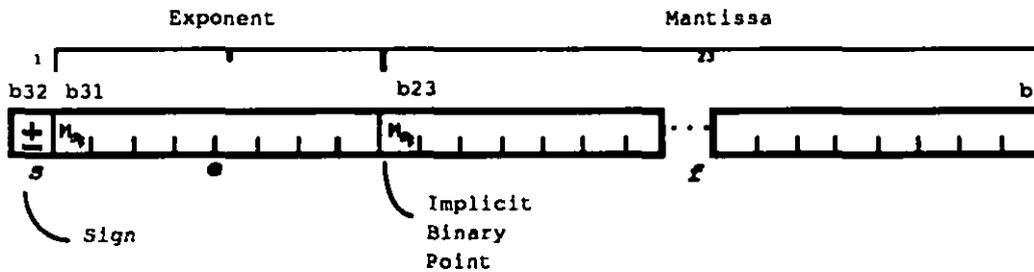
b. 32-bit format. For a 32-bit single format number, the value v is inferred from its constituents thus:

- (1) If $e = 255$ and $f \neq 0$, then v is NaN,
- (2) If $e = 255$ and $f = 0$, then $v = (-1)^s \infty$,
- (3) If $0 < e < 255$, then $v = (-1)^s 2^e - 127 (1 . f)$,
- (4) If $e = 0$ and $f \neq 0$, then $v = (-1)^s 2^e - 127 (0 . f)$ (denormalized numbers),
- (5) If $e = 0$ and $f = 0$, then $v = (-1)^s 0$ (zero).

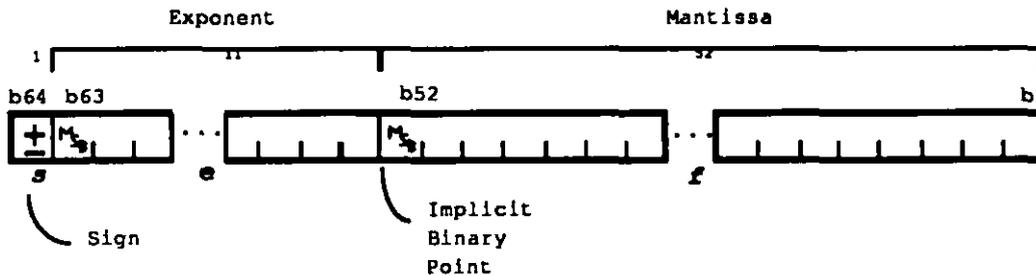
c. 64-bit format. For a 64-bit double format number the value v is inferred from its constituents, thus:

- (1) If $e = 2047$ and $f \neq 0$, then v is NaN,
- (2) If $e = 2047$ and $f = 0$, then $v = (-1)^s \infty$,
- (3) If $0 < e < 2047$, then $v = (-1)^s 2^e - 1023 (1 . f)$,
- (4) If $e = 0$ and $f \neq 0$, then $v = (-1)^s 2^e - 1023 (0 . f)$ (denormalized numbers),
- (5) If $e = 0$ and $f = 0$, then $v = (-1)^s 0$ (zero).

Note: the "." in equations (3) and (4) above corresponds to a decimal point.



IEEE Short Real Number Format Structure (32 bit)

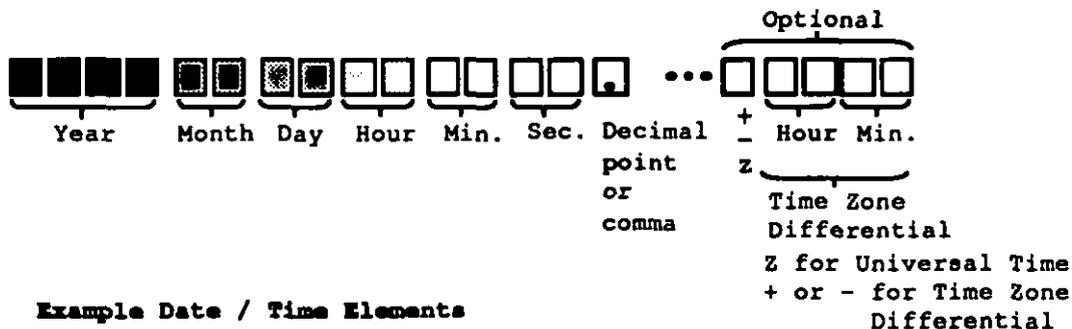


IEEE Long Real Number Format Structure (64 bit)

- Where :
- The sign bit is 0 for positive and 1 for negative.
 - The exponent is 8 bits long for short real numbers and 11 bits for long real numbers. The exponent is biased by 81 hexadecimal for short real numbers and 401 for long.
 - The remaining bits are the mantissa. Since the first significant bit is known to be set (since the mantissa is normalized), it is not stored. The length is 23 bits for short real numbers and 52 bits for long real numbers.

FIGURE 22. Real number syntax.

5.5.3 Date and time syntax. The generalized time data type consists of a string of international reference version (IRV) characters where the calendar date (as specified in ISO 8601) consists of a four-digit representation of the year, a two-digit representation of the month, and a two-digit representation of the day. This is followed by the time of day (as specified in ISO 8601) consisting of a string of digits containing a two-digit representation of the hour (based on the 24-hour clock), a two-digit representation of the minute, and a two-digit representation of the second, followed by a decimal point (or decimal comma) and an arbitrary number of digits of fractions of a second. This may be followed by the letter Z to represent coordinated Universal Time rather than local time, or be followed by a time differential from UT in accordance with ISO 8601 (see figure 23). In a fixed data field usage, date and time elements will be 20 bytes long,

**Example Date / Time Elements**

19870205160627. • Local time 6 minutes, 27 seconds
after 4 pm on 5 February 1987
- 19870205210627.Z • As above but Universal Time
(Greenwich)
- 19870205160627.-0500 • Local Time as above. The
local time is 5 hour time
behind UTC

FIGURE 23. Date and time syntax.

allowing for the specification of date and time with time zone differentials (or optionally fractional seconds). Unfilled digits will be filled with space characters. A null date time specification will consist of a string of space characters.

5.5.4 Text syntax. Two types of text strings are required in VPF. These are a basic string of ASCII text (used to encode items like universal labels). The other is a generalized text format that is used to handle place names and other information for any language in the world. A string of text may be taken from the IRV alphabet (defined in ISO 646). This standard is equivalent to ASCII. An international registry of all character code tables is also available (ISO 2375). The "Basic Text String" data element type would include any string of characters from the IRV alphabet. The "General Text String" encompasses any characters including accents, diacritical marks, special characters, and any other registered alphabet. This permits annotation on a cartographic display to be described in any language.

- a. Text strings. Textual information can be either variable length or fixed length. The null state of a variable-length text string is of zero length. The null state of a fixed-length text string requires that a specific code be selected. The CO character DEL (code table position 7/15) should be used as the padding character. The character code NUL (code table position 0/0) and a number of other CO control characters may have special meaning

on some computer systems and should not appear in any text strings. A NUL or a SUB (^Z) in a file is an end of file mark on some computers. Two types of text strings are supported in VPF:

1. Basic text string. These strings make use of characters only from the IRV (ASCII) primary code table and the subset of the CO table identified above.
 2. General text strings. These strings are composed of characters from any of the ISO registered code tables. Code extension (ISO 2022) is only required to handle written languages that are not based on the Latin alphabet. For languages (like English, French, German, or Spanish) that use the Latin alphabet, the IRV (ASCII) and the supplementary code table together with the identified subset of the CO set will be used.
- b. Code tables. All text is coded in terms of character sets. Particular character codes are identified by a code table arranged into rows and columns in which 94 character codes are assigned. A number of different character code tables are in use internationally and these code tables are registered with the International Standards Organization under ISO 2375. The basic international code table is the IRV alphabet (see figure 24).

The alphabetic code table is termed the graphic or "G" set. Another specialized code table, the control or "CO" set, is also defined. Some of the CO control characters are reserved for specialized use, such as transmission control in an asynchronous communications system or application level delimiting. VPF requires only the format effector CO characters, such as carriage return (CR) and line feed (LF), and the code extension characters escape (ESC), shift in (SI), and shift out (SO). The code extension characters allow extension to other alphabets as described below. Other CO characters are not used and have a null meaning. The IRV (ASCII) code table caters largely to the needs of the English language. For other Latin languages in which accented letters are used extensively, the ISO has recommended a supplementary character set for coded characters in text communication (ISO 6937). First, a nonspacing accent character is selected from the supplementary character set; then the accented character is selected.

' + e = é

row				column	b7	0	0	0	1	1	1	1		
b4	b3	b2	b1		b6	0	0	1	1	0	0	1	1	
					b5	0	1	0	1	0	1	0	1	
						0	1	2	3	4	5	6	7	
0	0	0	0	0					SP	0	@	P	`	p
0	0	0	1	1					!	!	A	Q	a	q
0	0	1	0	2					"	2	B	R	b	r
0	0	1	1	3					#	3	C	S	c	s
0	1	0	0	4					\$	4	D	T	d	t
0	1	0	1	5					%	5	E	U	e	u
0	1	1	0	6					&	6	F	V	f	v
0	1	1	1	7					'	7	G	W	g	w
1	0	0	0	8	BS				(8	H	X	h	x
1	0	0	1	9	HT)	9	I	Y	i	y
1	0	1	0	10	LF				*	:	J	Z	j	z
1	0	1	1	11	VT ^{Esc}				+	;	K	[k	{
1	1	0	0	12	FF				,	<	L	\	l	
1	1	0	1	13	CR				-	=	M]	m	}
1	1	1	0	14	SO				.	>	N	^	n	~
1	1	1	1	15	SI				/	?	O	_	o	

FIGURE 24. Latin alphabet primary code table (ASCII).

This supplementary code table (see figure 25) may be used with IRV (ASCII) or other national variants of IRV. In addition, the repertoire of all accented characters and diacritical marks (see figure 26) covers all Latin alphabet-based languages as represented by ISO 6937.

For languages based on other alphabets (such as Greek, Cyrillic, Chinese Hanzi, and Japanese Kanji or Katakana), independent code tables may be defined. These code tables are registered with the ISO and are assigned a final character code for use with a designated escape sequence. By use of these escape sequences, the current "in-use" code table may be switched.

The code table switching mechanism is specified by ISO 2022. The "in-use" code space is organized into rows and columns and divided into two areas, the "in-use" C area and the G area. The character ESC (escape) (position 1/11) is used as an introducer to sequences of codes which determine which code table is in use.

row					column								
b4	b3	b2	b1		b7	0	1	2	3	4	5	6	7
0	0	0	0	0		0	0	0	0	1	1	1	1
0	0	0	1	1		0	0	1	1	0	0	1	1
0	0	1	0	2		0	1	0	1	0	1	0	1
0	0	1	1	3		0	1	2	3	4	5	6	7
0	1	0	0	4									
0	1	0	1	5									
0	1	1	0	6									
0	1	1	1	7									
1	0	0	0	8									
1	0	0	1	9									
1	0	1	0	10									
1	0	1	1	11									
1	1	0	0	12									
1	1	0	1	13									
1	1	1	0	14									
1	1	1	1	15									

	NS	•	◻	-	Ω	Ɔ
	SP	±	`	'	Æ	æ
	€	²	'	ø	θ	σ
	£	'	^	©	®	ð
	\$	x	~	™	℥	℥
	¥	μ	-	Ƶ	◻	ı
	#	¶	~	~	ıı	ıı
	§	·	·	ı	ı	ı
	¤	+	¨	◻	ł	ł
	'	'	◻	◻	ß	ß
	"	"	•	◻	Œ	œ
	«	»	,	◻	ø	ø
	←	¼	◻	⅛	þ	þ
	↑	½	"	⅜	ƒ	ƒ
	→	¾	.	⅝	ŋ	ŋ
	↓	¿	˘	⅞	'n	SHY

FIGURE 25. Latin alphabet supplementary code table of accents, diacritical marks, and special characters.

a. International alphabets. The following is a list of the most common alternate alphabets from the international registry, together with their final character for the escape sequence used to designate.

- IRV (Latin) alphabet - 4/0
- UK variant of IRV - 4/1
- ASCII variant of IRV - 4/2
- Other set for use with variants of IRV - 6/12
- Katakana alphabet (Japanese) - 4/9
- Greek alphabet - 6/10
- Cyrillic alphabet - 4/14
- Extended Cyrillic alphabet - 5/1
- African languages alphabet - 4/13
- Arabic alphabet - 6/11

Basic Letter	Acute Accent	Grave Accent	Circumflex Accent	Diaeresis or Umlaut	Tilde	Caron	Breve	Double Acute Accent	Ring	Dot	Macron	Umlut Mark	Cedilla	Ogonek
aA	áÁ	àÀ	âÂ	äÄ	ãÃ		āĀ		āĄ		āĀ	āĀ		ąĄ
bB														
cC	ćĆ		ĉĈ			čČ				ċĊ			çÇ	
dD						đĐ								
eE	éÉ	èÈ	êÊ	ëË		ěĚ				ėĖ	ēĒ	ëË		ęĘ
fF														
gG	ğĞ		ĝĜ				ģĢ			ġĠ			çÇ	
hH			ĥĤ											
iI	íÍ	ìÌ	îÎ	ïÏ	ĩĨ					ıİ	īĪ			ıı
jJ			ĵĴ											
kK													ķĶ	
lL	ĺĹ					ľĽ							łŁ	
mM														
nN	ńŃ				ñÑ	ňŇ								
oO	óÓ	òÒ	ôÔ	öÖ	õÕ			őŐ			ōŌ	öÖ	õÕ	
pP														
qQ														
rR	ŕŖ					řŘ							ŗŖ	
sS	śŚ		ŝŜ			šŠ							șȘ	
tT						ťŤ							țȚ	
uU	úÚ	ùÙ	ûÛ	üÜ	ũŨ		ūŪ	űŰ	ųŲ		ūŪ	ûÛ		ųŲ
uU														
wW			ŵŴ											
xX														
yY	ýÝ		ÿŸ	ÿŸ								ÿŸ		
zZ	źŹ					žŽ				żŻ				

FIGURE 26. Usage of accents and diacritical marks.

- b. Chinese and Japanese alphabets. The Chinese and Japanese iconographic alphabets may also be designated. These character sets are special in that they require two consecutive bytes to index into over 8,000 entries. The designation sequences are given below:

Chinese Hanzi - ESC 2/4 4/1
 Japanese Kanji - ESC 2/4 4/2

5.5.5 Coordinate syntax. A coordinate specifies a position in the Cartesian unit coordinate space as a vectorial displacement from the origin of the coordinate space. A coordinate parameter value takes the form of an short or long floating point value.

5.5.6 Coordinate strings. Two types of coordinate strings are defined for use in VPF. These consist of coordinate tuples (pairs or triplets). All coordinate strings are constructed out of the number and coordinate formats defined in the previous subsections. A coordinate string consists of a sequence of coordinate parameter values corresponding to coordinate pairs.

6. NOTES

(This section contains information of a general or explanatory nature that may be helpful, but is not mandatory.)

6.1 Intended use. This standard is designed to define the methods and provide guidance for creating and using digital geographic databases in vector product format.

APPENDIX A

INTRODUCTION TO THE VPF DATA MODEL

10. GENERAL

10.1 Scope. This appendix provides information, discussion, and examples concerning the VPF data model. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions in section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 Introduction. VPF is a general, user-oriented data format for representing large spatially referenced (geographic) databases. VPF is designed to be used directly; that is, software can access the data without time-consuming conversion processing. VPF is designed to be compatible with a wide variety of users, applications, and products.

To achieve its generality and user orientation, VPF uses a georelational data model that provides a flexible but powerful organizational structure for any digital geographic database in vector format. VPF defines the format of data objects, and the georelational data model supporting VPF provides a data organization within which software can manipulate the VPF data objects.

The following paragraphs discuss in general the data model that serves as the basis of VPF. Section 40.2 discusses the basic concepts that form the foundation for all geographic data models. Section 40.3 describes the relational model, while section 40.4 describes the planar topology model.

40.2 Data model concepts. A model is a fundamental description of a system that accounts for all known properties of that system. The system is a view of geographic reality, or

APPENDIX A

information tied to specific locations in coordinate space. Since this particular model is stored in a computer, it is called a data model. A data model provides the most abstract representation of a system; it describes a collection of entities, including their relationships and semantics. The purpose of a data model is to define and capture a view of reality in a consistent and uniform manner. It provides a framework to visualize the structure and behavior of these entities in a system.

A model of a database requires three components: the definition of the data objects, data operations, and the rules of data integrity. These components are defined in the following three subparagraphs. Objects identify how the user perceives the data and its structure. The operations define how the user may manipulate the objects. Integrity rules bind the objects and operations, and establish a well-defined behavior that provides accurate information in a predictable manner. The fourth subparagraph deals with the purpose and functionality behind a database.

40.2.1 Data objects. Data objects identify how the user perceives the data and its structure. These objects also define the most primitive partitions of the data model architecture. For instance, an international database may contain a wide variety of objects concerning nations. The relevant objects could be areas, civil divisions, populations, resources, products, and water bodies. Relationships can be defined for these objects where populations and resources are grouped by civil divisions.

40.2.2 Data operations. Data operations define how a user may manipulate the objects; where, for instance, an object's attributes may be displayed, or new attributes could be defined, or, perhaps, new objects created. In most cases, an algebra is defined that accurately manipulates the data objects and binds the scope of operations within the data model. Classical database operations include retrieval, creation, deletion, and modification. More specific operations are defined for applications using the database.

40.2.3 Data rules. The rules of data integrity constrain the operations on objects in order to preserve overall stability. The goal is to prevent operations that yield corrupt, incorrect, or ambiguous results. Integrity rules constrain the set of valid states of databases that conform to the data model. These rules define the accuracy of the database.

40.2.4 Database purpose. One function of a database is to provide centralized control of operational data that is vital to an organization. A data model attempts to closely mesh the data objects, operations, and integrity rules into a cohesive system with optimal performance. The advantages of centralized database control are well established, and the use of a data model allows a

APPENDIX A

database to provide the following functionality to an organization.

- a. Consistency. The database will provide access to data in a formal manner. This will establish a consistent view of the data, enabling efficient data exchange.
- b. Simplicity. A basic objective is to provide an intuitive, straightforward, and understandable interaction between the user and the data.
- c. Nonredundancy. Duplication of data will be avoided wherever possible, especially when repetition provides little additional information.
- d. Multiple applications. The data model needs to support multiple end user applications because user views of the database will be different.
- e. Flexibility. A critical requirement of an effective database is the ability to accept new data. A database needs to have the dynamic flexibility to grow with the needs and requirements of its users.
- f. Integrity. The integrity rules should be defined in a consistent manner for all data objects and operations. The use of well-defined rules prevents operations that lead to data corruption and misinformation.

In summary, a data model provides a powerful approach to achieving optimum centralized control of critical data within a system. Using a variety of integrity rules and established operations upon defined data objects, the database provides users with the necessary tools for extracting data in support of many applications.

40.3 Relational data model concepts. The relational data model provides a powerful architecture for database design because of its ability to handle a wide variety of data and applications.

40.3.1 Relational data objects. The relational model uses simple tabular data structures to portray the data in a natural, well-defined manner. These data structures contain columns and rows, where columns define attributes, the values for which are taken from a range of data defined across a given domain. The content of the rows represents the actual data entities. The strength of the relational model is its ad hoc ability to establish meaningful data, transforming data into information as a function of user perspective. For instance, a relational table "ROADS" can be defined as having three columns: name, class, and structure. The rows that compose the ROADS table would contain distinct information about each road in each field in the

APPENDIX A

database. The six following properties distinguish relational tables from nonrelational data objects:

- a. Entries in columns must be single-valued; a field may not contain a list of attributes.
- b. Entries within a column must be of the same data type.
- c. Each row must be unique, and duplicate rows are not permitted.
- d. Columns may appear in any order.
- e. Rows may also appear in any order.
- f. Each column must have a unique name.

40.3.2 Relational data operations. The relational model supports eight set operations: select, project, product, join, union, intersection, difference, and division. Since the relational model is founded on set theory, the operations themselves are based on fundamental mathematical principles. These operations allow data objects to be manipulated and created in a specific manner, producing stable results.

40.3.3 Relational data rules. The integrity rules constrain operations performed on objects in order to preserve stability. The goal is to prevent operations that yield corrupt, incorrect, or ambiguous results. The entity integrity rule requires the entry of a null value in columns in relational tables for which the value is always known or understood. The referential integrity rule ensures that a foreign key referenced into another table stays within recognizable bounds. For example, a foreign key is not permitted that references a record number of 500 in a table that only contains 300 records. Additional, more subtle domain rules can also be defined to constrain entries in and operations on the database.

40.4 Plane topology model concepts. Conceptually, plane topology can be defined as a planar graph, where geographic reality is decomposed into a finite set of 0 cells (nodes), 1 cells (edges), and 2 cells (faces). This terminology is defined by an algebraic topology that establishes rules for decomposing continuous three-dimensional objects into representations of finite models. Once this topologic mapping has been performed, a system can be modeled in a way that permits more complex relationships between objects to be established.

The purpose of topology is to capture and retain knowledge concerning a cell's spatial and thematic relationships with its neighboring cells. For a topologic model to be valid, these relationships must remain constant regardless of changes in scale,

APPENDIX A

shape, or size. With topology embedded in a data model, very useful relations can be established, such as adjacency and connectivity. Topologic and geometric relationships (such as size, angle, and shape) provide powerful resources that allow geographic reality to be fully modeled.

40.4.1 Topologic objects. Plane topology extends graphic models of nodes and edges to the development of a more powerful and expressive model that contains spatial relationships. In addition to metric capabilities (distance, shape, or size), topology determines spatial neighbor relations. By defining more rigorous and complex relationships in the data model, the true properties of a system can be more effectively represented. Various mathematical models are available. In addition, simpler models can be used to create more complex ones. The most simple is the graphic model. More complex and useful surface-based models are built upon the graphic, using topology to capture additional analytical information.

A graphic model represents geometric information based on node and edge primitives. This model provides the base for other models that define more complex relationships. The node primitive is composed of a location in an established coordinate space. Most representations are two dimensional (x,y) or three dimensional (x,y,z). An edge is composed of a minimum of two nodes, with more details concerning linear or spline interpolation between the end points.

Because of the complexity of geographic information and the limitations of a graphic model, plane topology provides a better model for defining relationships. The use of a planar topology model (based on the two-dimensional manifold), for instance, maps more concisely in the two-dimensional space that current computing systems can manage. VPF provides four distinct levels of topology: full planar topology (level 3); a linear planar graph (level 2); a nonplanar linear graph (level 1); and no topology defined, indicating a geometric model (level 0). VPF uses the notion of topology as a constraint to enforce integrity rules upon the feature definitions. As the entities require fewer topological relationships, the rules can be relaxed. For instance, if linear features in a transportation network are being modeled, then the requirement of full planar topology may be relaxed because it is not necessary.

Plane topology defines relations between cells without modifying the underlying geometry. The concept of a cell can be visually retained in graphic models, but only allows one view of the data. Topology establishes a framework that provides more information for analysis. For instance, in figure 27, the edges 1, 2, 3, 4, and 5 are grouped together into face 'a.' Another face known as 'b,' including the edges 5 through 12, can be defined. Topology

APPENDIX A

defines relationships on each 0-, 1-, and 2-cell in a model. Face 'b' is defined to be "left-of" edge 5; edge 7 can be defined to

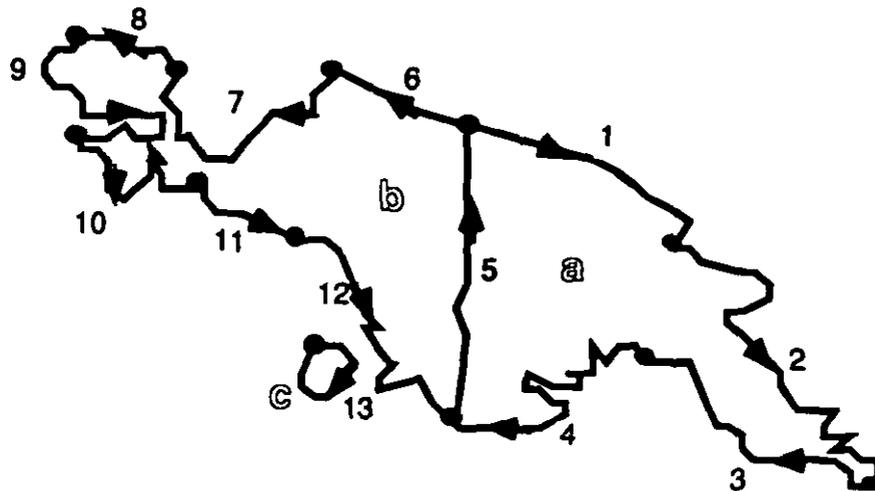


FIGURE 27. The definition of faces.

follow edge 6 in a cycle. This topologic information provides the power to determine orientation, adjacency, and connective relationships between objects.

The concept of topology held by the geometric primitives is carried upward to features and their associated thematic information. An area feature is usually labeled, or contains information pertaining to the enclosed region. For instance, an area can have a category (soil class, surface material type) or a numeric value (population size, number of airports). Topology is used to provide operations and information to distinguish between these thematic objects. Thematic relationships can exist between features without requiring the primitive geometry. For instance, a set of the islands in the Pacific Ocean (Oahu, Maui, Hawaii, Molokai, etc.) can be defined as different features with differing geometric primitives, but they can be related to one another as the Hawaiian Islands.

40.4.2 Topological operations. Topological operations are based on the single notion of adjacency—that is, if two objects are next to each other, it is necessary to maintain the adjacent relationship between them. To distinguish the topological aspects from the geometric aspects of geography, we are only concerned with whether two objects, A and B, are adjacent to each other and not with whether A is bigger than B, or one is to the north of the other, or the length of their common boundary.

Many complex topological operations can be derived from adjacency alone. In the georelational data model implemented for VPF, two

APPENDIX A

topological operations are paramount: boundary and coboundary. For example, an edge has a start node and an end node; the nodes are the boundary for the edge. The edge, in turn, is the coboundary of the node. Of course, the coboundary of a node can have more than one edge if many edges meet at a node. Similarly, faces have edges as boundaries. The coboundaries of edges are maintained in the left and right faces.

40.4.3 Topological rules. The integrity rules of the topological model are contained in the definition of the objects themselves. A plane model restricts itself to planar geometry, where all entities must lie in the same plane. In addition, all faces must be mutually exclusive and not overlapping. These constraints allow the objects to be defined in context and allow operations to be performed in a consistent manner. While these rules may seem to restrict the system model, they define the data model's domain, taking advantage of the underlying structures. By restricting the faces to be constructed of nonoverlapping regions, powerful set operations can be applied to the objects (such as union, intersection, or join).

40.5 The VPF georelational data model. VPF uses a combination of the relational and the planar topologic data models to provide a hybrid model for geographic data management, analysis, modeling, and display. The georelational model provides the data structure foundations for a spatial database, and software provides the rules and operators that manipulate topology, geometry, and relational objects in the form of tables. Whenever an operation requires thematic information, the use of relational and topologic table operations are used to supply the result. If the operation is spatially related, geometry and topology together will be used. This triad of categories (geometry, topology, and relational tables) provides a robust database architecture.

VPF adheres to the georelational data model, but only defines the objects and the data structures that compose the objects. The georelational operations and algebra are not part of the standard, but rather are implemented in software. Every VPF object is described in the form of a relational table, composed of columns defining the syntax of each field and rows that contain the actual data.

APPENDIX B

WINGED-EDGE TOPOLOGY

10. GENERAL

10.1 Scope. This appendix provides information, discussion, and examples concerning winged-edge topology. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions of section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 Winged-edge topology. Winged-edge topology is an essential part of the VPF data model. The function of winged-edge topology is to provide line network and face topology and also to maintain seamless coverages across a physical partition of tiles. The following sections define the components of winged-edge topology, the algorithm used to traverse the winged-edge network, and cross-tile topology.

40.2 Components of a winged edge. Winged-edge topology uses three specific components (columns) on an edge primitive table to provide connectivity between nodes, edges, and faces. Given level 1, 2, or 3 topology, the edge primitive will contain specific columns for each topologic level. As shown by figure 28, there are three topologic constructs: node, edge, and face information on each edge. These constructs are formally defined in section 5.3. A brief summary of the definition is repeated below.

- a. Node information. Each edge will contain a start node and an end node column. This topologic information is used to define an edge direction (digitizing direction).

APPENDIX B

- b. Edge information. Right and left edges connect an edge to its neighbor edges (thus the term "winged edge"). The right edge is the first edge connected to the end node that is encountered when cycling around the node in a counterclockwise direction. The left edge is the first edge connected to the start node that is encountered when cycling around the node in a counterclockwise direction.

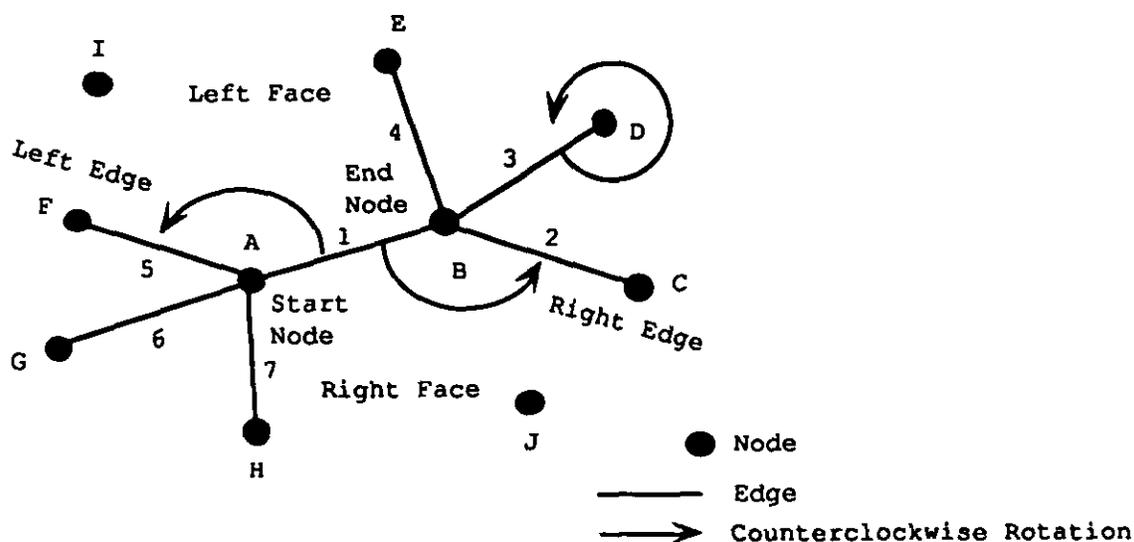


FIGURE 28. Winged-edge components.

- c. Face information. With level 3 topology specified, each edge will contain a left and right face. Left and right face are determined solely by the edge direction. This information allows an edge to know its neighboring faces.

40.3 Winged-edge algorithm. Given the definition of a winged edge, every coverage containing faces and edges is created in the same way. With the enforcement of a planar topologic model, a consistent navigation algorithm can be applied.

Figure 29 depicts a collection of faces and accompanying edges. To navigate a face, the following algorithm is used:

- a. Determine which face to draw. Determining which face to draw would typically be driven by the selection of area features that have the attributes desired; then key through the face and ring tables associated with the area.

An example of an area feature table consistent with figure 29 is shown in table 58.

APPENDIX B

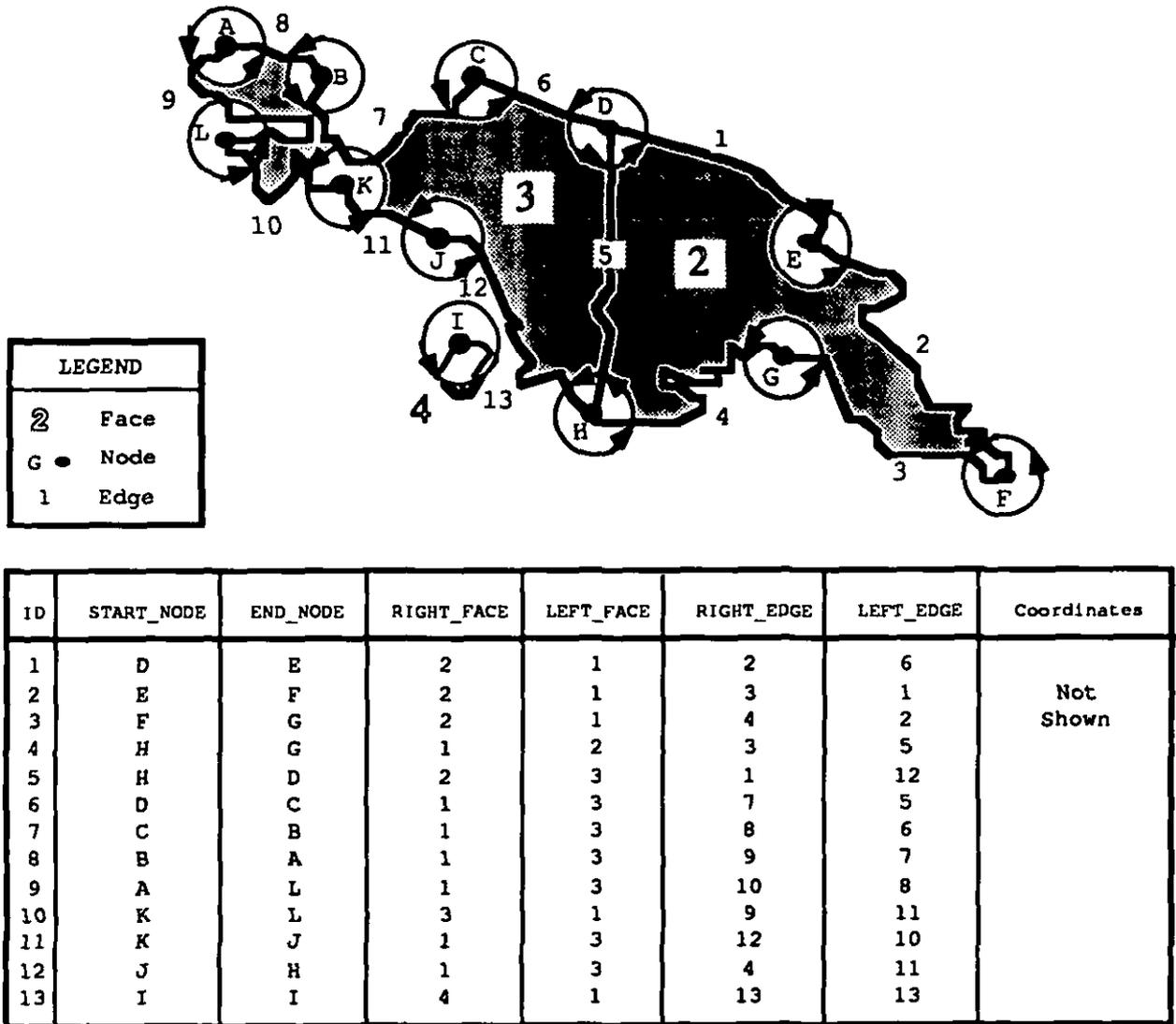


FIGURE 29. Winged-edge example, with drawing completely contained within tile 95.

TABLE 58. Sample area feature table for figure 29.

ID	TILE_ID	FAC.ID	<attribute 1>...< >
.	.	.	.
.	.	.	.
436	95	3	1 ...
.	.	.	.
.	.	.	.
.	.	.	.

APPENDIX B

Table 58 identifies tile 95 with face 3 as the primitive corresponding to area feature 436. The face table for tile 95 for figure 29 (table 59) yields the following:

TABLE 59. Sample face table for figure 29.

ID	*.AFT_ID	RING_PTR
.	.	.
3	436	7

The ring table for figure 29 (table 60) yields the following:

TABLE 60. Sample ring table for figure 29.

ID	FACE_ID	START_EDGE
.	.	.
7	3	12

- b. Now identify the start edge (in this case, 12).
- c. Travel to the left edge to trace the left face; the right edge for the right face. Because face 3 is the left face of edge 12, read the left edge record (edge 11). Edge 11 leads to edge 10, edge 10 to edge 9, edge 9 to edge 8, edge 8 to edge 7, edge 7 to edge 6, edge 6 to edge 5, and edge 5 to edge 12.
- d. Edge 12 is the start edge, so the cycle is complete.

The same figure can be navigated as a linear network, regardless of the existence of face topology (ring list, left face, right face). Assume that a line feature table exists for this example and that every line feature has an attribute column that is used to determine the correct edge selection in the network. The value of a line attribute will determine the traversal criteria and also determine when it will end. The starting point and endpoint of the traversal are dependent upon the user's application. A network can be traversed with various strategies. The example below illustrates a depth first search. The algorithm is as follows:

APPENDIX B

- a. Locate current edge with user application (edge 12).
- b. Read end node of current edge and gather all edges incident at the node.
- c. For each of the edges incident at the node, read the attribute value from the associated line feature. Does the attribute have the desired value? If so, continue with step e.
- d. Go to step c. Repeat with next edge.
- e. Decision. Has the network been completely traversed? If so, exit with complete network. If not, go to step b.

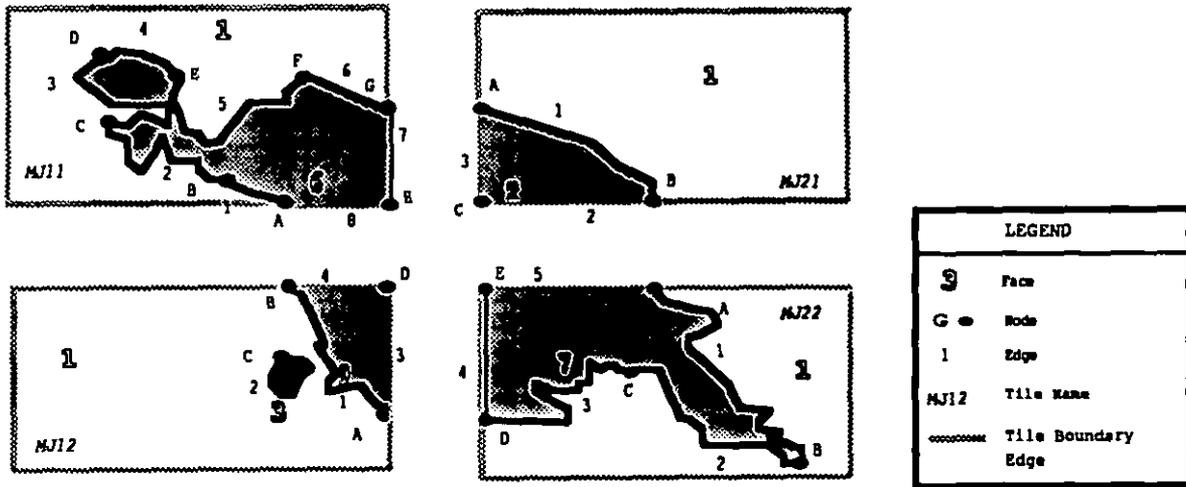
40.4 Cross-tile topology. Network navigation using the winged edge can be extended to cross over physical tile partitions, if they exist in the coverage. By using the information in the previous examples, it becomes possible to introduce cross-tile constructs. Assume that figure 29 has been intersected with tile boundaries and that the new coverage in figure 30 has been created (with generalized edges along edge 5 in figure 29). The following discussion concerns retrieving the island separated into the four tiles shown in figure 30. Only two of the tile edge tables have been included for the sake of brevity. Note that the face (face 1) must be traversed counterclockwise if the entire island is to be traversed. Since all features, with respect to the universe face, are inner rings, the traversal method must be applied counterclockwise. Navigating clockwise would only retrieve the equivalent of face 2 in figure 29.

When creating the cross-tile topology, observing the following rules is necessary:

- a. If any edges are inserted into the coverage to maintain face topology, these edges are labeled tile boundary edges; they are not legitimate features and do not participate in cross-tile topology. Only features and their composing primitives can cross into other tiles.
- b. When tiles are crossed, an edge can only connect to one other edge in another tile. The two edges must have the same node coordinates.
- c. When two or more edges in another tile are encountered, the external edge which is the first counterclockwise neighbor is chosen.

When the face algorithm described in the first paragraph above is used, the face retrieval proceeds as follows:

APPENDIX B



TILE MJ21

ID	Start Node	End Node	Right Face Face, Tile, ExFace	Left Face Face, Tile, ExFace	Right Edge Edge, Tile, ExEdge	Left Edge Edge, Tile, ExEdge	Coordinates
1	A	B	2,0,0	1,0,0	2,MJ22,1	3,MJ11,6	Not Shown
2	B	C	2,0,0	1,0,0	3,0,0	1,0,0	
3	C	A	2,0,0	0,MJ11,6	1,0,0	2,0,0	

TILE MJ22

ID	Start Node	End Node	Right Face Face, Tile, ExFace	Left Face Face, Tile, ExFace	Right Edge Edge, Tile, ExEdge	Left Edge Edge, Tile, ExEdge	Coordinates
1	A	B	7,0,0	1,0,0	2,0,0	5,MJ21,1	Not Shown
2	B	C	7,0,0	1,0,0	3,0,0	1,0,0	
3	D	C	1,0,0	7,0,0	2,0,0	4,MJ12,1	
4	D	E	7,0,0	0,MJ12,9	5,0,0	3,0,0	
5	E	A	7,0,0	0,MJ21,2	1,0,0	4,0,0	

FIGURE 30. Cross-tile edge example.

- a. Start with tile MJ21, edge 1. Read the left edge; notice that choosing either of two edges is possible. Choose to cross into tile MJ11, edge 6.
- b. Chain from edge 6, 5, 4, 3, 2, and 1 within tile MJ11.
- c. From edge 1 in MJ11, go across to tile MJ12, edge 1.
- d. From edge 1 in MJ12, cross tiles into tile MJ22, edge 3.
- e. Chain through edges 3, 2, and 1.
- f. From edge 1 in MJ22, cross into tile MJ21, edge 1.
- g. When the end of the face cycle is reached, exit.

APPENDIX C

FEATURE CLASS RELATIONS

10. GENERAL

10.1 Scope. This appendix provides information, discussion, and examples concerning feature class relationships. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions of section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 Overview. One of the most important parts of designing a VPF database is the implementation of a conceptual feature class model into feature class tables. The heart of this implementation concerns the relationships with the feature class tables and their composing primitive tables.

Depending on the design, the relationships between the features and primitives can be one-to-one (1:1), one-to-many (1:N), many-to-one (N:1), or many-to-many (N:M). This appendix is intended to help designers implement their conceptual feature classes with two major implementation constraints: software performance, tiled coverages, and indexing.

40.1.1 Software performance. Software performance is a leading consideration in a VPF database design. VPF is designed to support interactive software use. In order for software to respond optimally, the designer must determine the intended use of the database product. If intended for interactive use, the designer should follow any guidelines to improve performance. If the database is strictly for exchange purposes and not interactive use, a simpler implementation is recommended.

40.1.2 Tiled coverages. Tiled coverages require a triplet id column on the feature table or its associated join table. This triplet id foreign key allows the relationship to tiled

APPENDIX C

primitives. Alternatively, the tile and primitive id fields of the triplet id can be maintained as separate columns in the tables. Depending on the feature table type, table 61 lists the column names that shall be used as the join column name.

TABLE 61. Feature table join column definitions.

Area Feature	FAC_ID
Line Feature	EDG_ID
Point Feature	END_ID
Point Feature	CND_ID
Text Feature	TXT_ID

40.1.3 Indexing. VPF is designed for interactive use, and the following indexing recommendations apply to help software users achieve these goals.

40.1.3.1 Thematic indexes. It is recommended to use a thematic index on any feature or join table that contains the primitive tile information. For instance, the primitive column (FAC_ID, EDG_ID, CND_ID, END_ID, TXT_ID) should have an index created on the TILE_ID field in the column. This can improve software searches based on features located per tile.

40.1.3.2 Spatial indexes. All primitive tables in a VPF database should carry associated spatial indexes. The software performance can be improved significantly.

40.2 Feature and primitive table relationships. The following subsections provide many implementations of the feature and primitive table relationships. There are other options available, but other implementations may require product-specific software and are not advised.

Each subsection is based on the five types of relationships (1:1, 1:N, N:1, N:M, and complex). Within each subsection, there are a variety of implementations available, depending on tiling, performance, and index support. The following criteria are used to help determine why one implementation would be used instead of another.

40.2.1 Performance. The performance rank can be one of the following:

- a. Optimal. This is the best design for the software use. It usually implies a direct relation between two tables using their row ids.
- b. High. The resulting performance from this implementation is good, but one column in a table

APPENDIX C

has a many relation. A thematic index improves the performance from fair to high.

- c. Fair. Performance is not high due to the use of a join table or join columns without thematic indexes.
- d. Poor. Use of a join table or join columns and no indexes to increase performance to fair or high.

40.2.2 Orientation. There are two types of orientations:

- a. Product. This implementation is product-oriented, meant to be used with software. For purposes of this appendix, statements regarding product performance are based on experience accessing VPF data from a CD-ROM using a PC-386 computer.
- b. Exchange. This implementation is not optimized for product use; the data is meant to be converted into another format for use.

40.3 One-to-one relationships. When there is a 1:1 relationship between the feature and primitive tables, the relations are straightforward, except for tiling. The row id of the feature table is used as a foreign key into the row id of the primitive table. This is also true in reverse.

40.3.1 Untiled coverage. The untiled 1:1 design (figure 31) is the simplest implementation. Its performance is optimal and serves both product and exchange uses.

Performance: optimal
Orientation: product and/or exchange

APPENDIX C

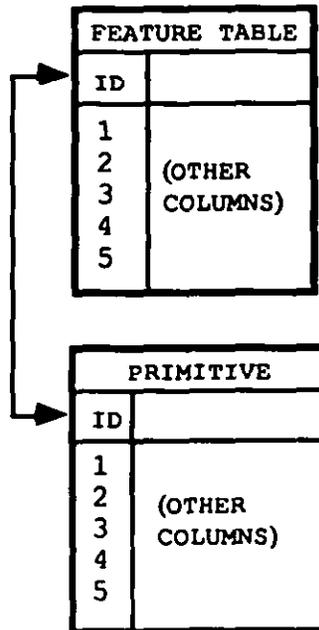


FIGURE 31. Implementation of an untiled 1:1 coverage.

40.3.2 Tiled coverage. The tiled 1:1 design (figure 32) is required to maintain direct relations between one feature and one primitive stored in only one tile. The performance of this implementation is optimal going from the feature to its primitive, but poor going from the primitive back to the feature.

Performance: poor
Orientation: exchange

APPENDIX C

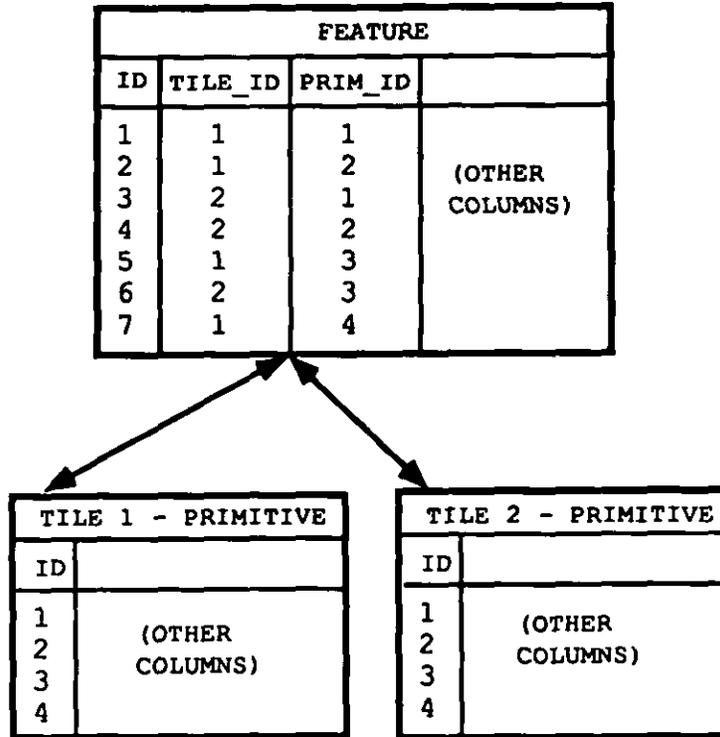


FIGURE 32. Implementation of a tiled 1:1 coverage.

APPENDIX C

40.3.3 Tiled coverage with feature performance. Adding a thematic index to the implementation described in 40.3.2 (figure 33) improves performance.

Performance: fair
 Orientation: product

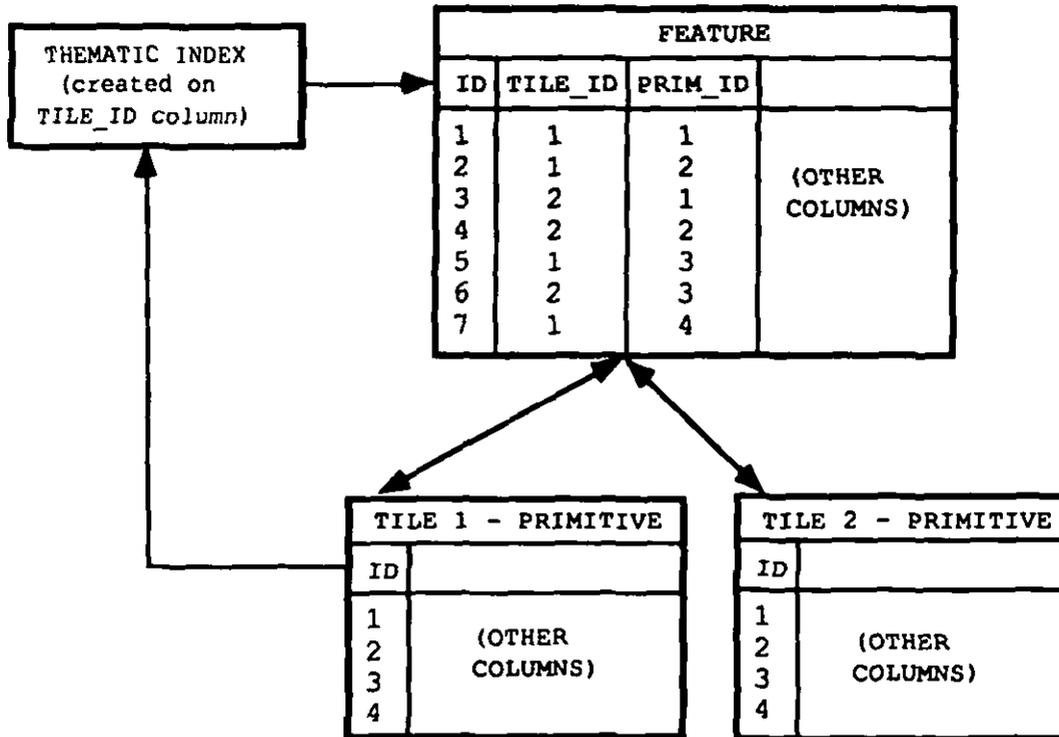


FIGURE 33. Implementation of a tiled 1:1 coverage with a thematic index.

APPENDIX C

40.3.4 Tiled relationship with primitive performance.
 Adding a FEATURE_ID column to the primitive tables to serve as pointers (figure 34) improves performance to high.

Performance: high
 Orientation: product

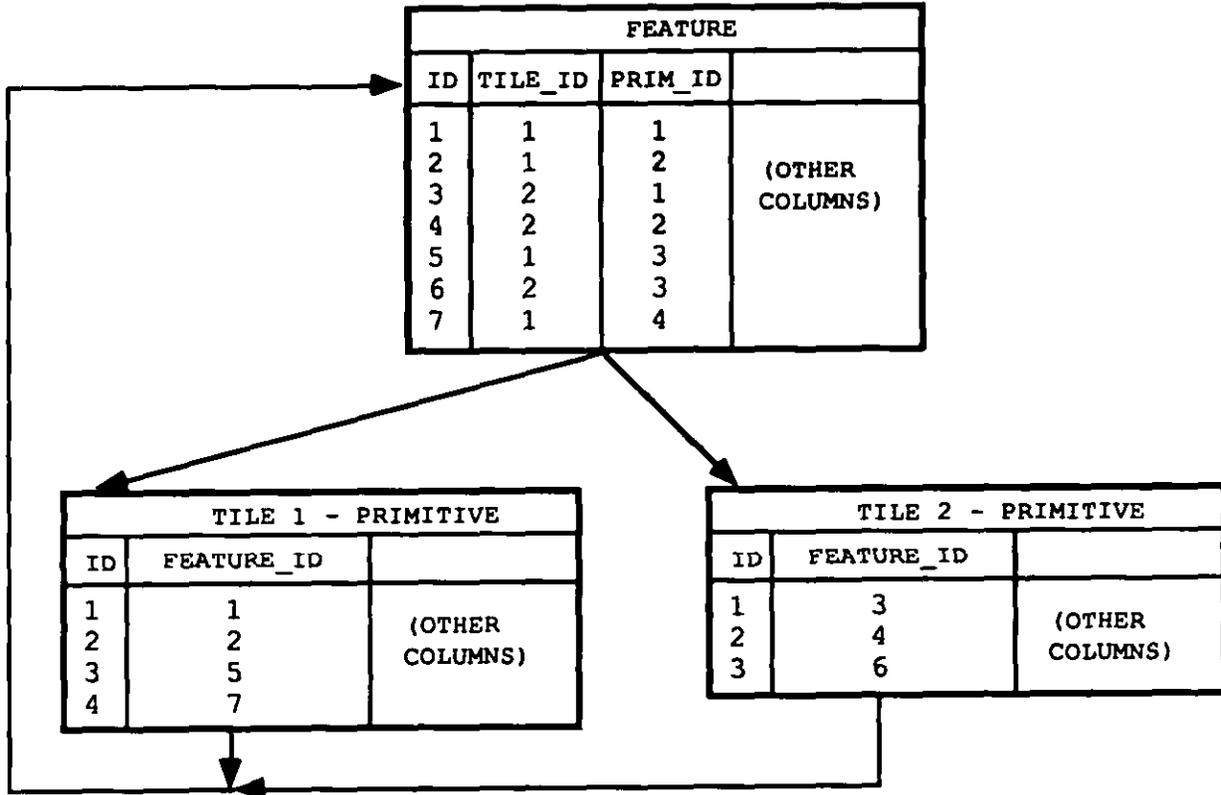


FIGURE 34. Implementation of a tiled 1:1 coverage with FEATURE_ID columns added to the primitive tables.

APPENDIX C

40.3.5 Tiled relationship with feature and primitive performance. Adding the thematic index to the implementations previously stated (figure 35) provides the optimal performance for a 1:1 tiled feature class.

Performance: optimal
 Orientation: exchange

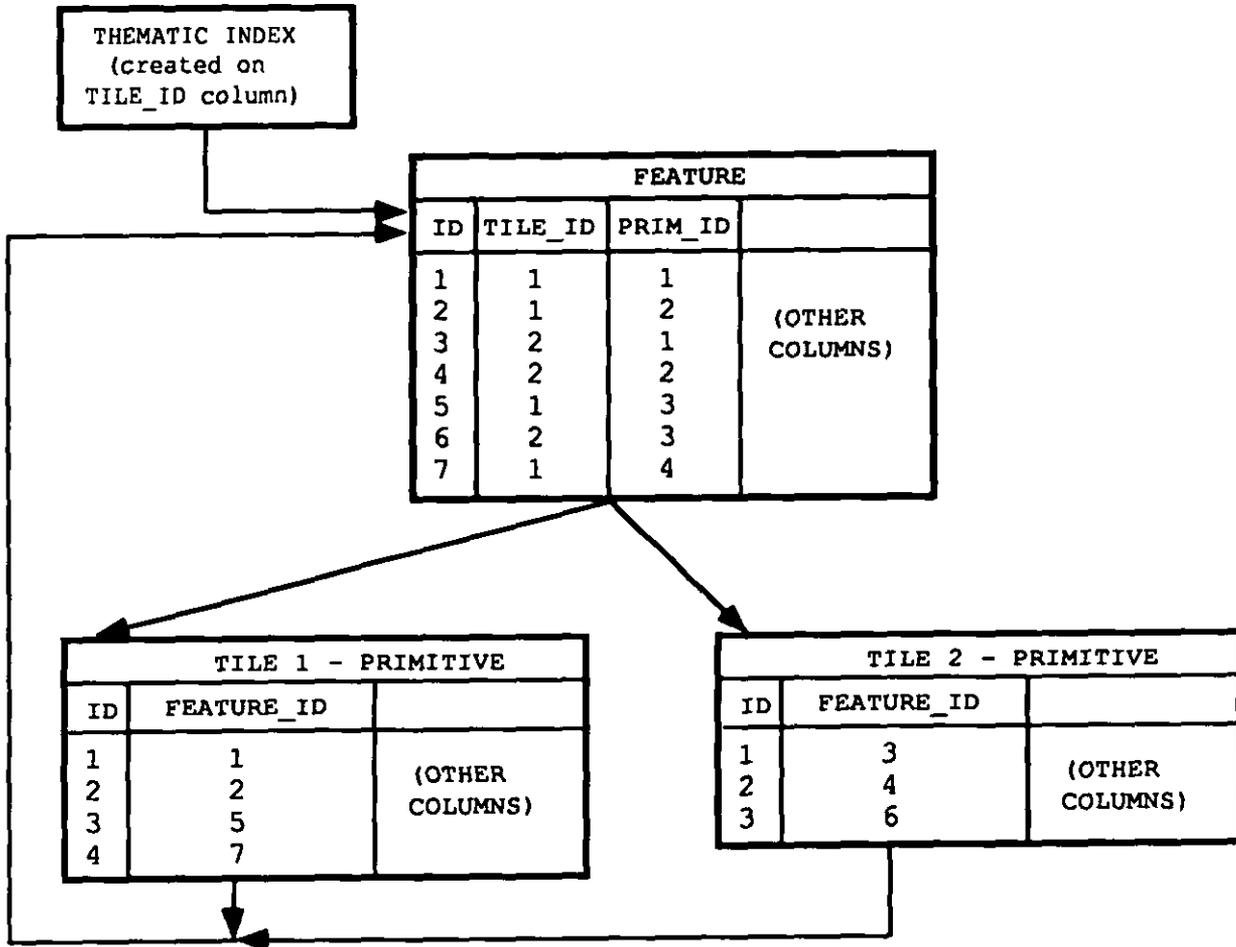


FIGURE 35. Implementation of a tiled 1:1 coverage with FEATURE_ID columns in the primitive tables and a thematic index.

APPENDIX C

40.4 One-to-many relationships. The next relation type is one in which one feature is composed of many primitives. This relation requires an additional join column for the FEATURE_ID in the primitive table or possibly the use of a join table. The row id of the feature relates to the join column of the primitive or join table. This join column is composed of the feature table name, appended with "_ID".

40.4.1 Untiled coverage. The simplest approach to the 1:N relationship is to add a FEATURE_ID column to the primitive table (figure 36). This allows for optimal performance going from primitive to feature, but makes for fair to poor performance going the opposite direction.

Performance: fair
Orientation: exchange

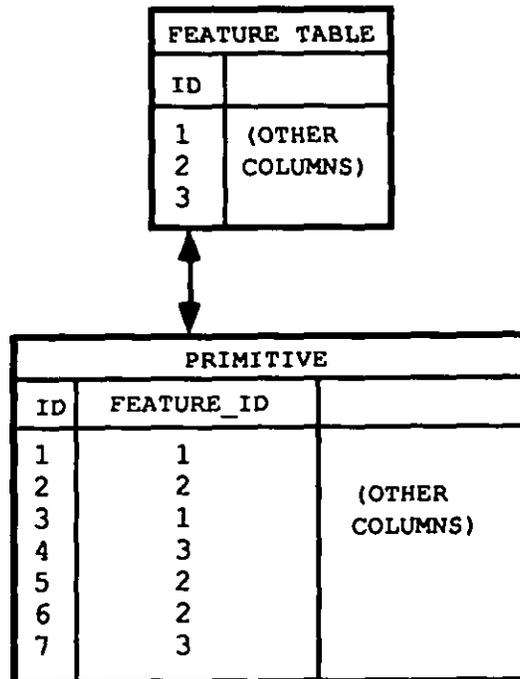


FIGURE 36. Implementation of a 1:N untiled coverage that includes a FEATURE_ID column.

APPENDIX C

40.4.2 Untiled coverage with feature performance. By including a thematic index on the FEATURE_ID column on the primitive (figure 37), the performance is now optimal for this relationship.

Performance: optimal
Orientation: product

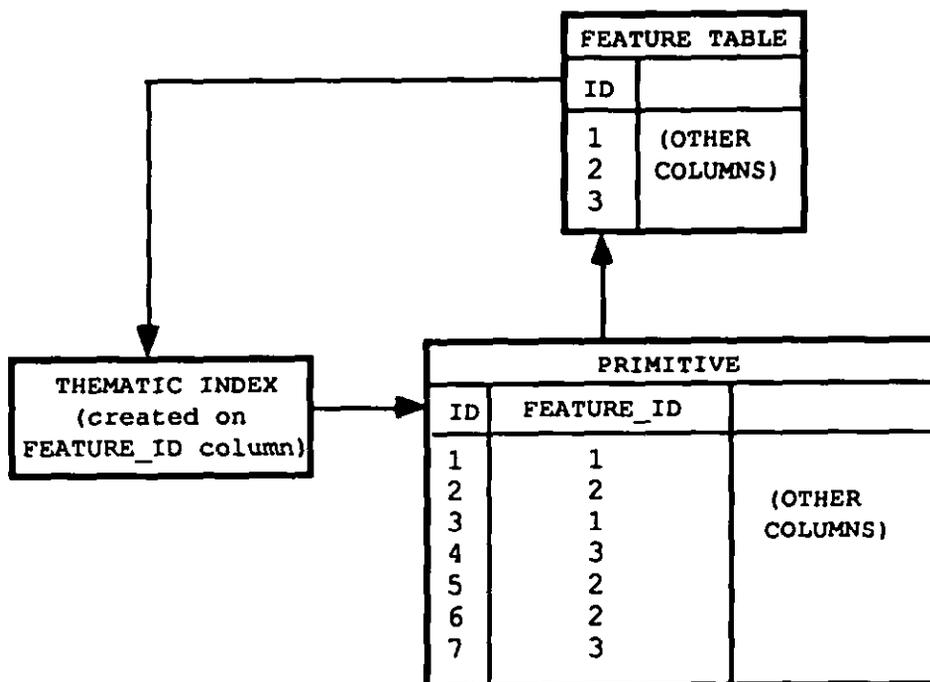


FIGURE 37. Implementation of an untiled 1:N coverage that includes a FEATURE_ID column and a thematic index.

APPENDIX C

40.4.3 Tiled coverages. The addition of tiles in a 1:N relationship makes for a complex implementation. The implementation shown in figure 38 is conceptually clean but will perform poorly with software.

Performance: poor
Orientation: exchange

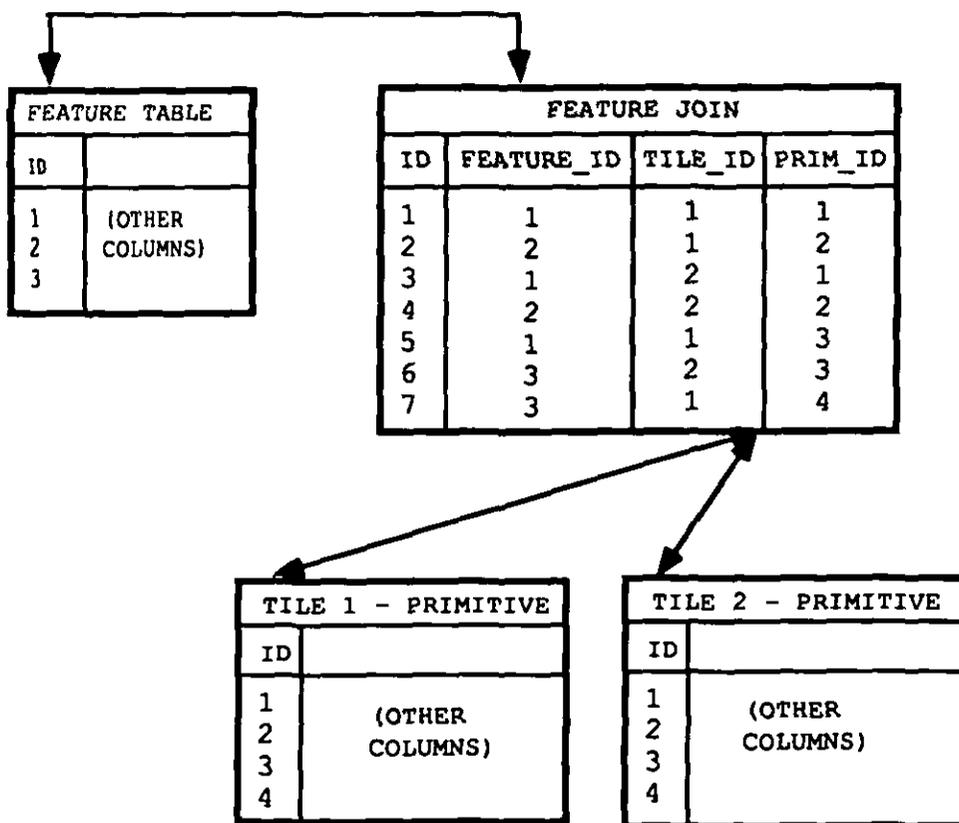


FIGURE 38. Implementation of a tiled 1:N coverage.

APPENDIX C

40.4.4 Tiled coverages with primitive performance. The addition of a FEATURE_ID column to the primitive tables (figure 39) provides optimal performance from the primitive to the feature. Performance from the feature to the join table to the primitive is poor.

Performance: fair
 Orientation: product

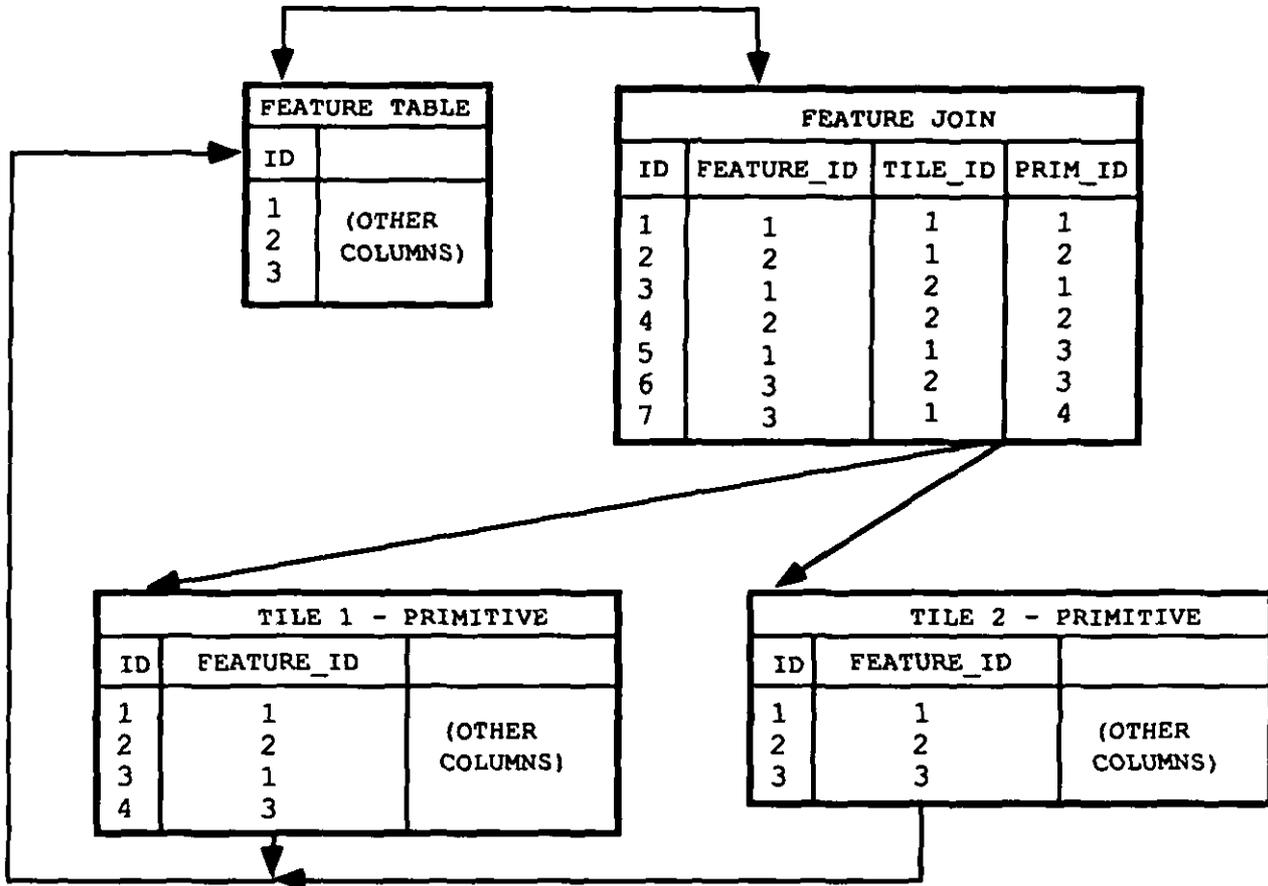


FIGURE 39. Implementation of a tiled 1:N coverage that includes FEATURE_ID columns.

APPENDIX C

40.4.5 Tiled coverage with feature and primitive performance. The addition of thematic indexes to the join table (figure 40) allows for the optimal performance for tiled 1:N relations.

Performance: optimal
 Orientation: product

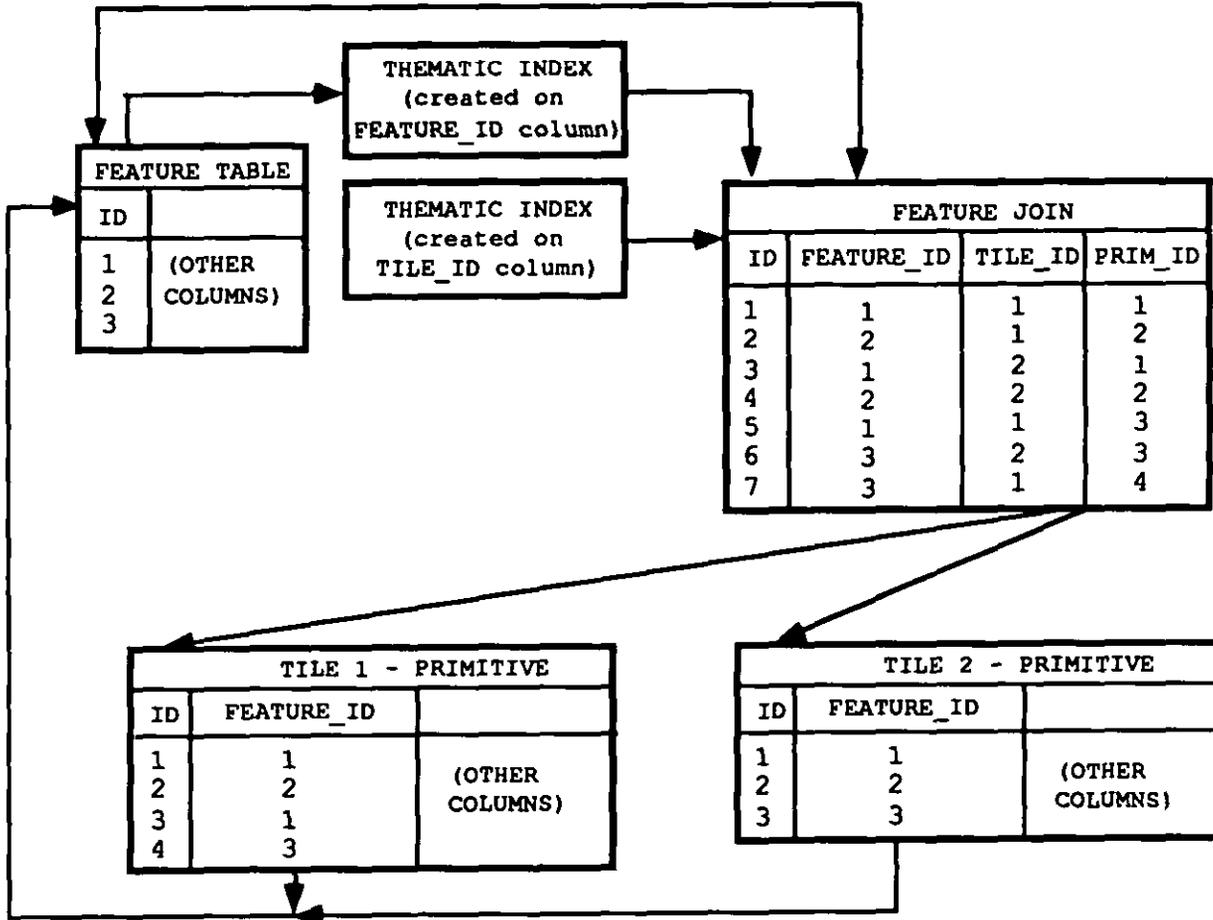


FIGURE 40. Implementation of a tiled 1:N coverage that includes FEATURE_ID columns and two thematic indexes.

APPENDIX C

40.5 Many-to-one relationships. When there are coincident features on the same primitive (for instance, when a river and a border contain the same edge), the relationship is thought of as N:1. A join column must be used on the feature or a join table must be used.

40.5.1 Untiled coverages. With the implementation shown in figure 41, performance is optimal going from the feature to the primitive, but is poor returning to the feature from the primitive.

Performance: fair
Orientation: exchange

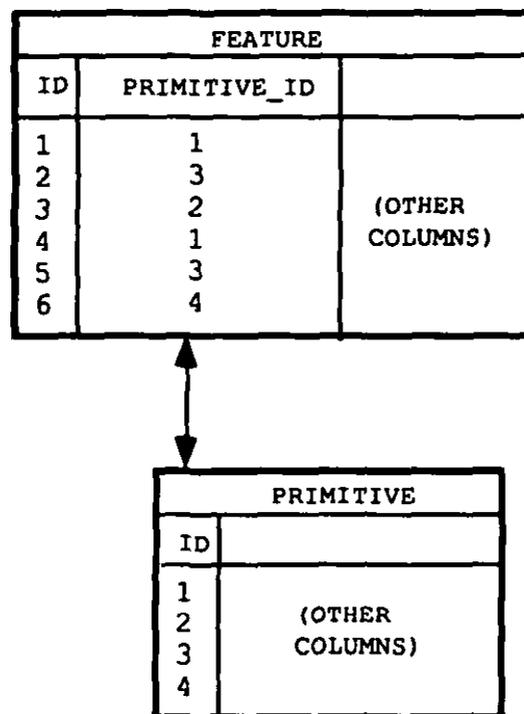


FIGURE 41. Implementation of an untiled N:1 coverage.

APPENDIX C

40.5.2 Untiled coverages with feature performance. The introduction of a thematic index on the PRIMITIVE_ID column in a feature table (figure 42) provides the optimal performance found in an N:1 relationship without tiles.

Performance: optimal
Orientation: product

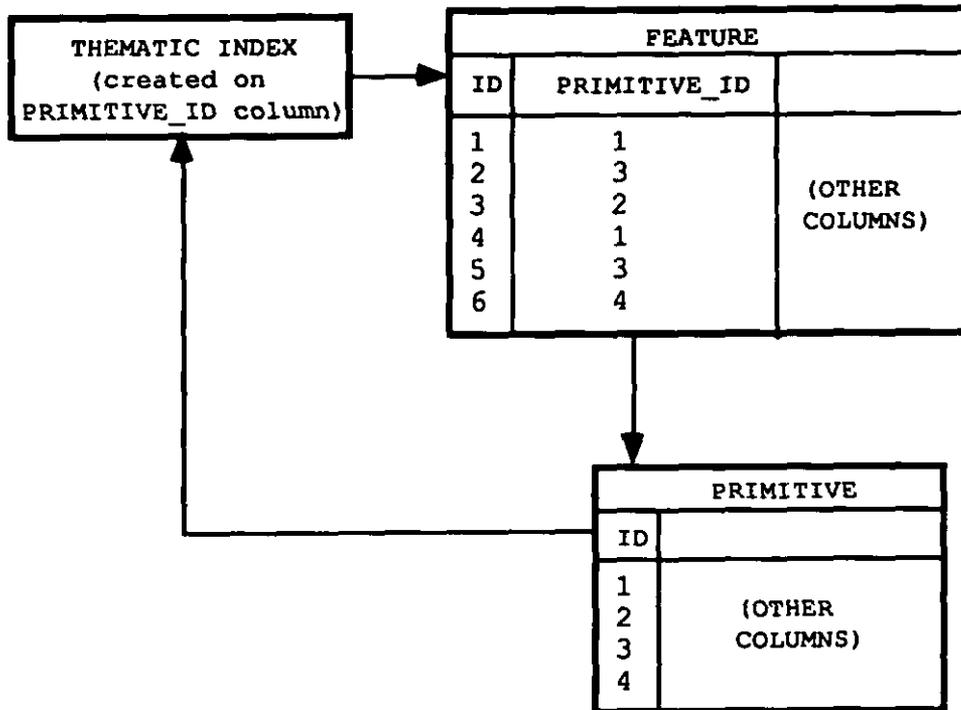


FIGURE 42. Implementation of an untiled N:1 coverage with a thematic index.

APPENDIX C

40.5.3 Tiled coverages. The introduction of tiles does not impact performance compared with an untiled N:1 relation (figure 43). Still, performance going from primitive to feature is doubly complex because of the search on the primitive and tile id in the feature.

Performance: poor
 Orientation: exchange

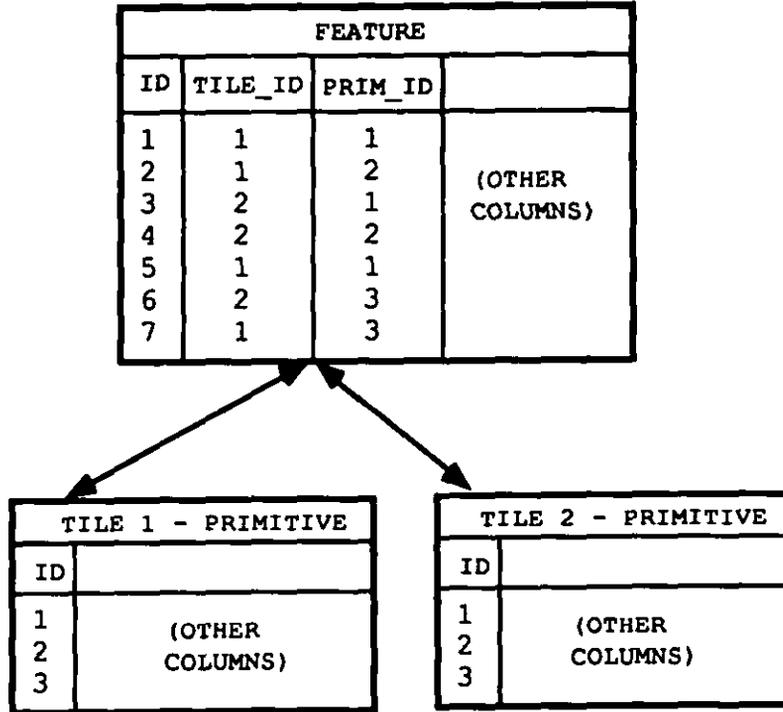


FIGURE 43. Implementation of a tiled N:1 coverage.

APPENDIX C

40.5.4 Tiled coverages with feature performance. The addition of two thematic indexes (figure 44) makes performance optimal for the given constraint of tiles.

Performance: optimal
 Orientation: product

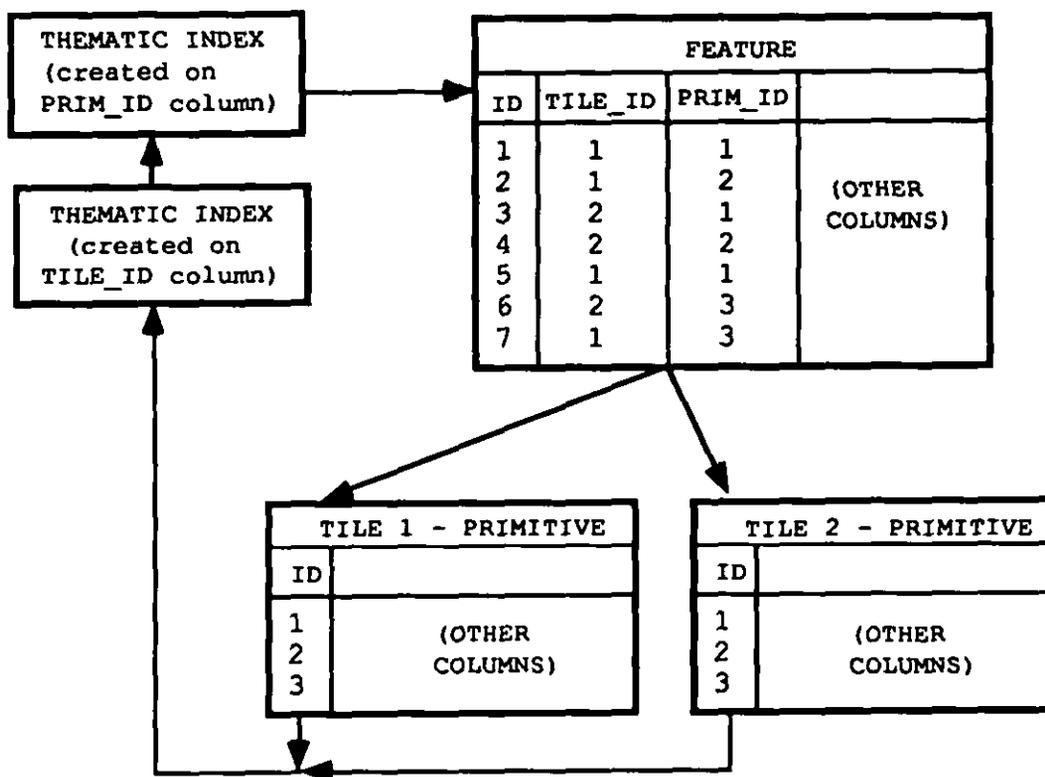


FIGURE 44. Implementation of a tiled N:1 coverage that includes two thematic indexes.

APPENDIX C

40.6 Many-to-many relationship. In many-to-many (N:M) relationships, many features can relate to many primitives and the reverse. Many features can relate to one primitive, and many primitives can relate back to one feature. This relation is established using a join table.

40.6.1 Untiled coverages. The implementation of this relation is complex (figure 45), and it is hard to improve performance.

Performance: poor
Orientation: exchange

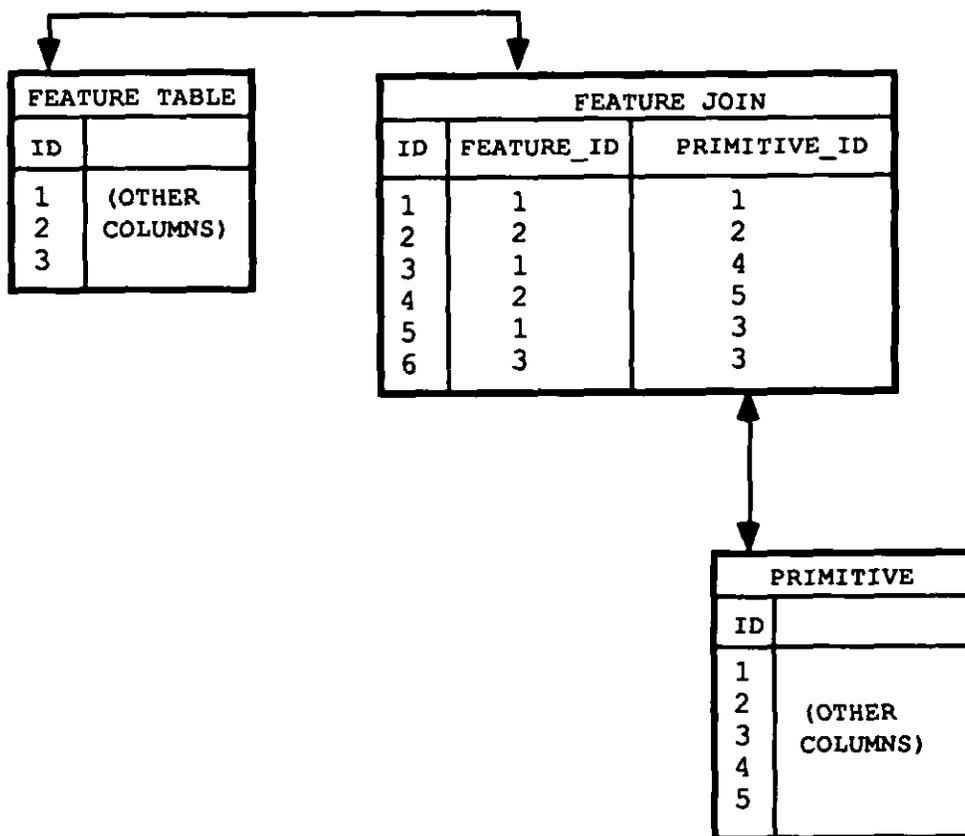


FIGURE 45. Implementation of an untiled N:M coverage.

APPENDIX C

40.6.2 Untiled coverages for performance. Given the complexity of the relationships, the addition of two thematic indexes (figure 46) allows the optimal performance possible.

Performance: optimal
 Orientation: product

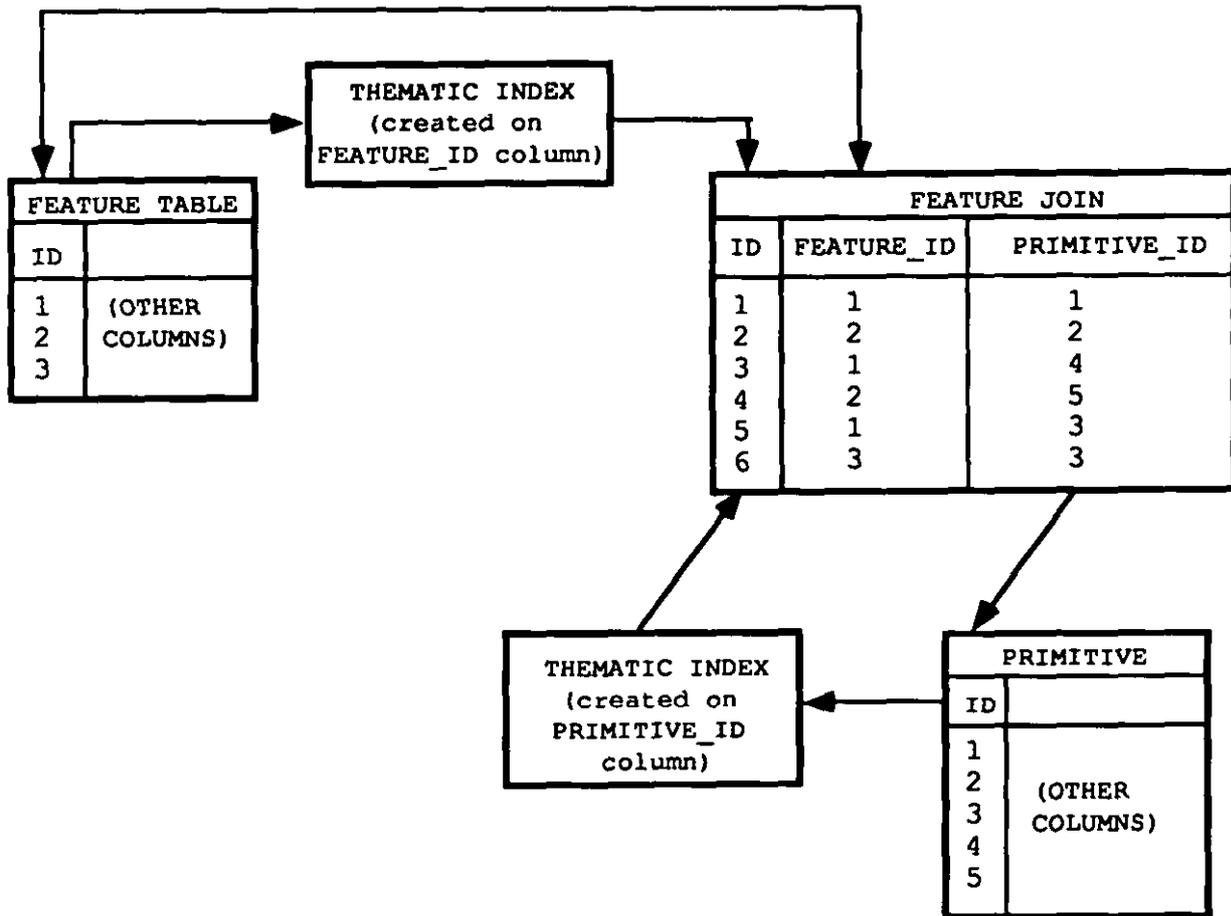


FIGURE 46. Implementation of an untiled N:M coverage with two thematic indexes.

APPENDIX C

40.6.3 Tiled coverages. The introduction of tiles (figure 47) provides poor performance without the use of thematic indexes.

Performance: poor
Orientation: exchange

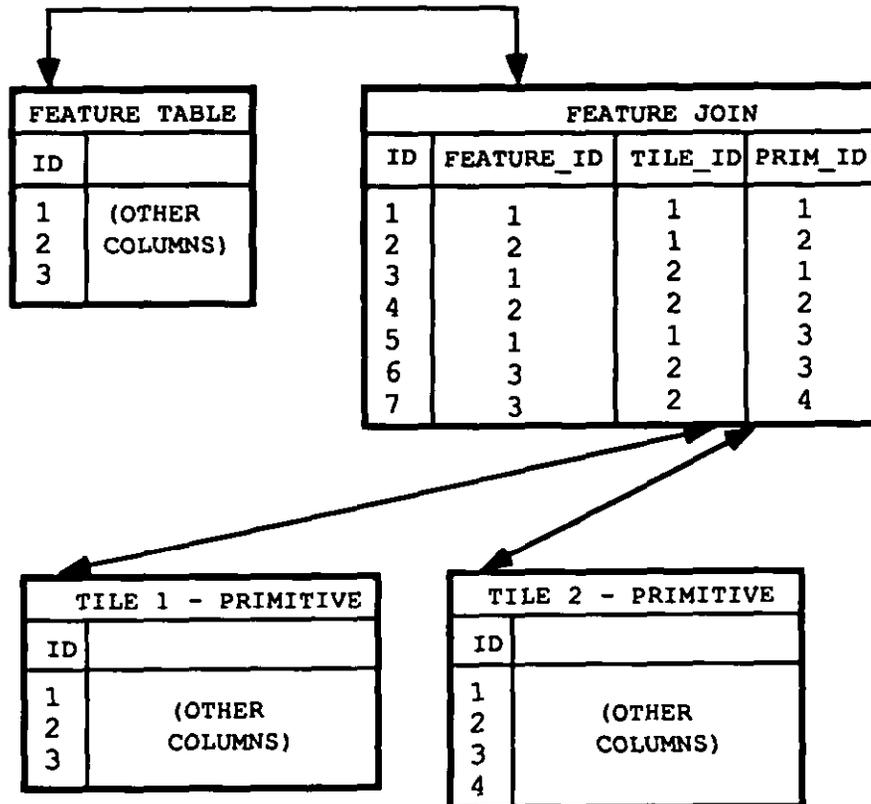


FIGURE 47. Implementation of a tiled N:M coverage.

APPENDIX C

40.6.4 Tiled coverages with primitive performance. The introduction of feature columns (figure 48) improves the performance going from primitives to features, but the feature to join table to primitive is still poor.

Performance: fair
Orientation: product

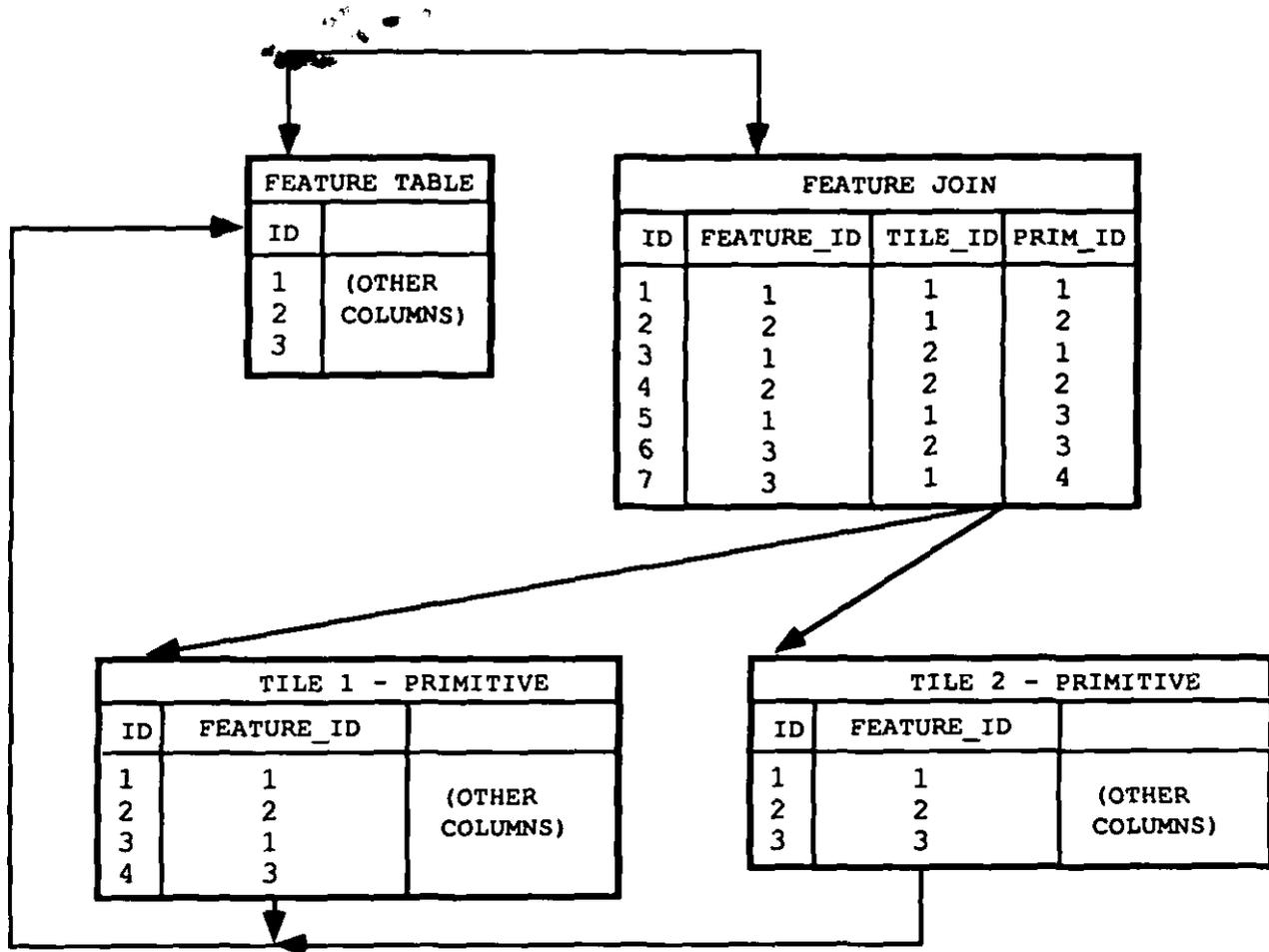


FIGURE 48. Implementation of a tiled N:M coverage that includes FEATURE_ID columns.

APPENDIX C

40.6.5 Tiled coverages with performance. The introduction of thematic indexes on the join table (figure 49) makes this implementation optimal for tiled N:M relations.

Performance: optimal
 Orientation: product

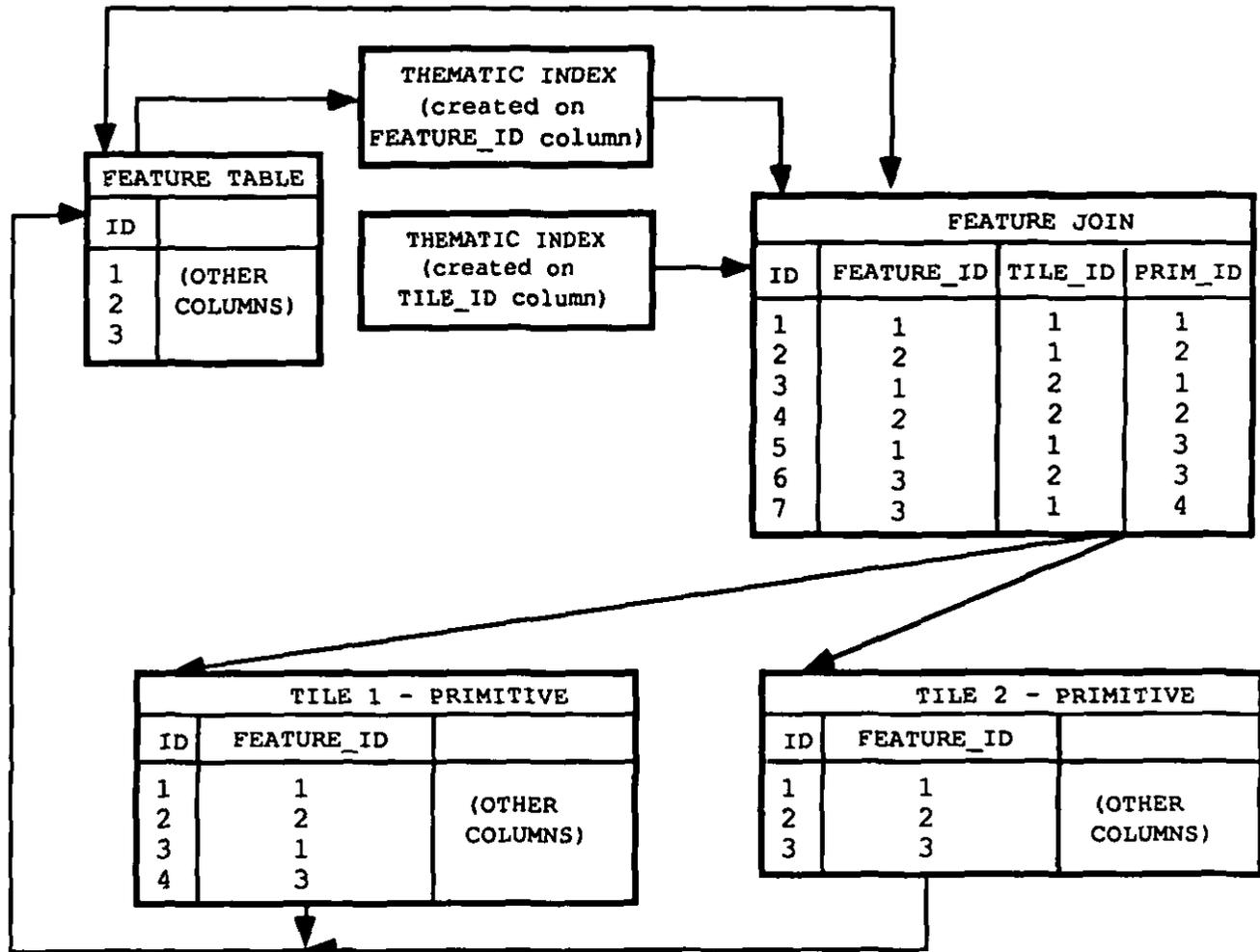


FIGURE 49. Implementation of a tiled N:M coverage that includes FEATURE_ID columns and two thematic indexes.

APPENDIX C

40.7 Complex feature relationships. A complex feature may be constructed from simple features only or from other complex features. This forms a hierarchical feature relationship. The need for complex features arises when a group of features requires different attributes than that of other features.

40.7.1 One complex feature, simple features in separate tables. If one complex feature is made up of several simple features, each record in the complex feature relates to one record in each simple feature table. The implementation shown in figure 50 is useful when not all simple features are part of a complex feature or when the complex feature is created after the simple features.

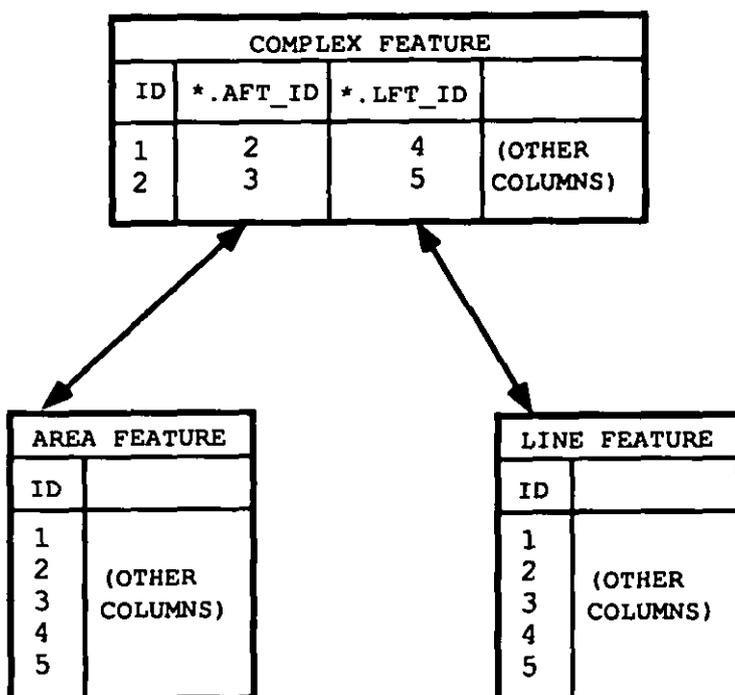


FIGURE 50. Implementation of a complex feature made up of simple features in separate tables.

APPENDIX C

40.7.2 One complex feature, many simple features in one table. One complex feature may be made up of many simple features in one feature table. The implementation shown in figure 51 requires complex features and simple features to be created at the same time. Performance can be improved by adding a thematic index on the CFT_ID column in the feature table.

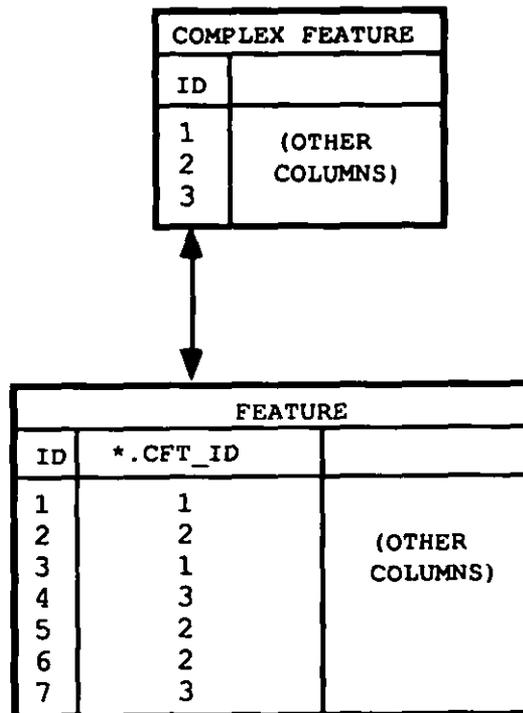


FIGURE 51. Implementation of a complex feature made up of simple features in one feature table.

APPENDIX C

40.7.3 Many complex, many simple features. Many complex features may be made up of many simple features in one feature table. The implementation shown in figure 52 requires complex features and simple features to be created at the same time. Performance can be improved by adding thematic indexes on both the columns in the join table.

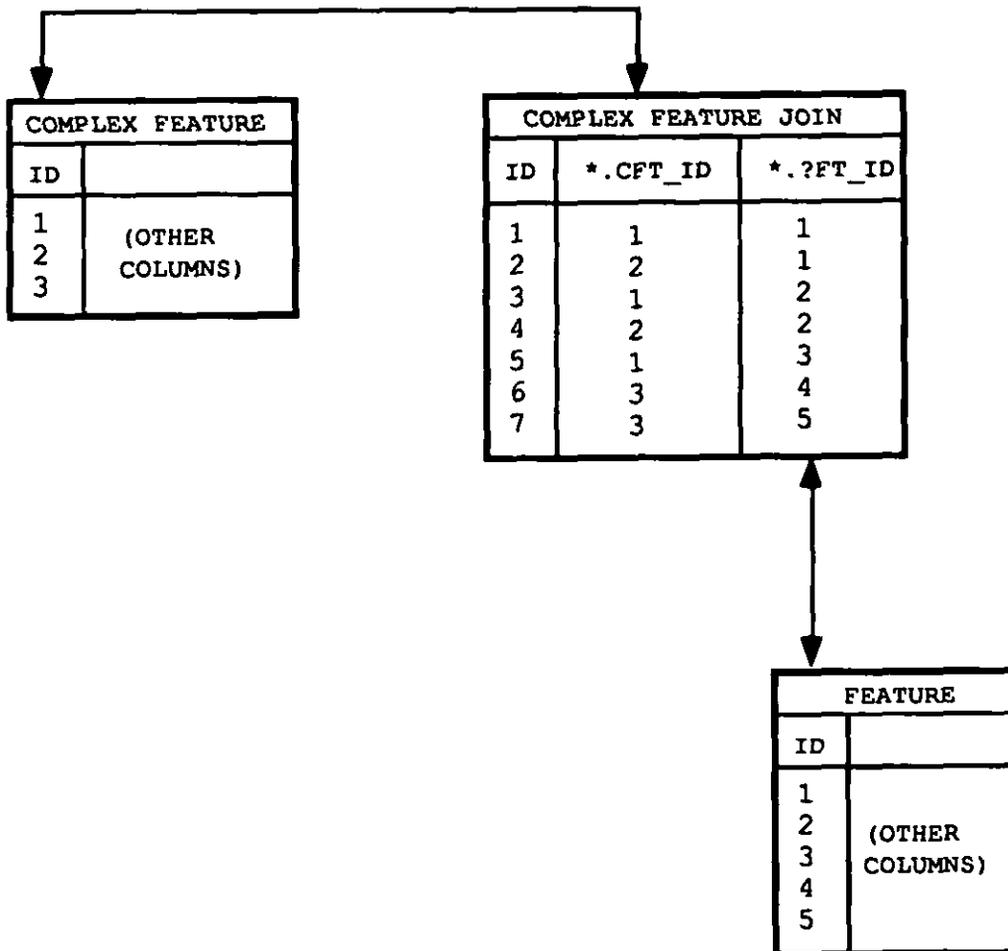


FIGURE 52. Implementation of a complex feature relationship in which many complex features are made up of many simple features in one feature table.

APPENDIX D

TILING

10. GENERAL

10.1 Scope. This appendix provides information and discussion concerning tiling for a VPF database. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions of section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 Rationale. Global scale databases inevitably consist of large amounts of data. In processing geographic data, entire files often need to be managed in memory, imposing a definite limit on database size. Tiling is the method used to break up geographic data into spatial units small enough to fit within the limitations of the desired hardware platform and media. Thus, the actual tile size is a product-specific question dependent upon the minimum hardware configuration and distribution media. The following paragraphs describe how VPF implements tiling to subdivide a database.

40.2 Cross-tile topological primitives. One shortcoming of past tiling implementations occurs when primitives are split up into different files, which also removes the topological connectivity of the feature. If a lake feature's primitive is split into two separate tiles, the topological connection between the primitives of the lake is lost. They will still appear together when both tiles are drawn on the screen, but any analysis that tries to follow the original connectivity of the lakeshore will have to do a lot of extra processing and searching. It would thus be advantageous for the primitives of the tiled lake shoreline to refer to each other. There is a need for primitives that cross tile boundaries to refer both to the tile boundary itself (in order to maintain tile topology) and to its cross-tile continuation in order to simplify retrieval of the original

APPENDIX D

feature. VPF meets this need by referring to edges using a triplet id that contains an internal reference to a boundary or edge within the current tile; when appropriate, the triplet id also contains an external reference to an edge in a neighboring tile.

40.3 Feature classes. Tiling introduces a constraint on feature classes as well as primitives. Tiling is a low-level implementation issue related to hardware and storage limitations, and should have no effect upon conceptual structures like feature classes. Unfortunately, when primitives are broken up into separate tables for tiling, their corresponding attributes are broken up as well. VPF resolves this by maintaining the attribute tables and feature class tables unbroken and structuring them so that they can be processed sequentially rather than all at once, thus obviating the need to break them up to meet hardware limits. These tables are stored within a coverage, and the actual file subdivisions representing the tiles appear as directories underneath the coverage. The problem concerns connecting a primitive, now stored in one of many smaller files, to its attributes, now stored in a single large file. This is done in accord with standard relational design rules by adding a column for each feature class onto the primitive file, containing the row id from the master table that has the attributes for that primitive. If five feature classes are derived from the primitive topological layer, then five extra columns will be added to that primitive table. Adding another column for each feature class could make for a very large table since there are 295 FACS feature classes. Not all feature classes apply to all three primitive dimensions or to all products, or to all coverages, but for vertically or thematically integrated data the number could still easily approach 100. The reason for a pointer back to the feature is merely for performance issues. Please refer to feature class construction issues in appendix C.

VPF handles tiling and data partition problems at the primitive level by means of the triplet id to maintain cross-tile topology and the extra feature class columns on the primitive to maintain links to the more logically consistent single attribute table. There is also the need to maintain the tiles themselves and to store reference data about the tiles, their size, scheme, and so forth. This is accomplished with the tile reference coverage.

40.4 Tile reference coverage. The tile reference coverage is a polygon coverage representing only the tiles. No other data besides tile boundaries and tile labels is used. This is stored as a separate coverage at the library level and acts as a graphic index to the tiling scheme, showing all of the tiles and only those tiles in the library, their names, and their relation to each other. The area feature table of this coverage plays a very important role. Since it can store attributes about each tile, it can be used to hold data density figures, tile data volume,

APPENDIX D

summary contents for each feature class, and other metadata to assist in managing the database at a coarse tile-by-tile level. The tile size, actual tile layout, and handling of text that crosses tile borders are not addressed in this standard since they are all product-specific questions.

APPENDIX E

DATA QUALITY

10. GENERAL

10.1 Scope. This appendix provides information, discussion, and examples concerning data quality issues in a VPF database. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions of section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 VPF data quality. This appendix describes basic strategies for storing data quality (DQ) information within VPF databases. The following subsections discuss general data quality concepts, implementation of the data quality table, and data quality coverages within VPF.

40.2 General concepts. The multilevel structure (i.e., primitive, feature class, coverage, library, and database) of a VPF database affects the strategies used to store DQ information. DQ information is often available at varying levels of specificity, from individual feature attributes to expressions of quality germane to an entire database. Therefore, it is necessary for the VPF data producer to determine the appropriate level in the hierarchy for the various types of DQ information present. When developing an implementation strategy, the data producer should also review the available DQ information within the context of coverage (thematic) associations as well as spatial extent, as these will influence the use of standard data quality tables and/or the development of separate data quality coverages.

When DQ information is stored at multiple levels in a VPF database, lower level information always takes precedence over that at higher levels. Data producers should account for this when compiling DQ information to be stored in VPF. For example, a data producer may make a general statement at the library level

APPENDIX E

that a coverage has been derived from a range of sources, and specify at the primitive level what a given feature's exact origins are. Alternatively, some variations may be organized by simple spatial divisions, such as source map boundaries. In such a case, feature level source information would be highly repetitious, and a data quality coverage might be most effective.

40.3 Data quality tables. The data quality table (table 47) is a device for storing standard and nonstandard types of DQ information. Standard information is explicitly defined within the DQ table. Nonstandard information is that which is not accounted for in the DQ table and must be stored outside the standard fields. The DQ table can be implemented on the database, library, or coverage level, depending upon the uniformity of the affected data with respect to known DQ characteristics. Key characteristics are source, positional accuracy, attribute accuracy, logical consistency, completeness, resolution, and lineage. All of these characteristics can be documented within standard DQ table fields, with the exception of lineage.

40.3.1 Lineage. Lineage information must be stored within the DQ table narrative file, which should be given the name "LINEAGE.DOC." The lineage file is an important component of the DQ documentation system, since the data producer must inevitably make key decisions affecting the data's fitness for use that cannot be described in standard fields. Lineage information may also change significantly from coverage to coverage, even when all of the data is derived from a single source. At a minimum, the lineage file should contain information on processing tolerances, interpretation rules applied to source materials, and basic production and quality assurance procedures. All lineage information available through the source should also be incorporated here.

40.3.2 Placement of DQ table. When implementing DQ tables, the user must determine at what level within the VPF hierarchy the DQ table(s) should be established. This will vary depending upon the specific nature of the DQ information. For example, entire libraries originating from a single source may be best served with a single table at that level, augmented by a series of data producer-defined tables implemented at the coverage level to document more specific information. The DQ table can also be implemented on multiple levels simultaneously, with general (broad) information provided at the higher levels, and progressively more detailed information specified at the lower levels. At a minimum, some form of DQ information should be present at the database or library levels to provide an overview of characteristics and to describe the techniques used to store DQ information within the database as a whole. With respect to VPF tables in general, the data producer is not restricted to the standard DQ table. The table can be augmented as needed with producer-defined related files within the hierarchy.

APPENDIX E

40.4 Data quality coverages. Data quality coverages delineate spatial variations in DQ information across a database or a library by assigning unique quality characteristics to areas. Within this context, the database and library levels must be viewed as having distinct spatial extents to which attributes can be assigned, while data quality coverages are created to document spatial variations on a more detailed level. DQ coverages are very useful as visual reference data that allow data producers to capture anomalous data behavior within the context of known spatial variations. They are, by definition, level 3 topology coverages. They can be physically stored at any level (above the feature level) within the VPF hierarchy, independent of the level of information being represented. The manner in which the DQ information is structured within the coverage will be dependent upon the relationships between spatial variations, data sources, and lineage information. Edges, as well as areas, can also be attributized within DQ coverages, for example, to document the interaction of disparate data sets that are not well reconciled. A more specific example is where contour lines from different data sources meet along a common edge and fail to match positionally. In this case, the data producer could document this discrepancy as an attribute of the seam (edge) between the adjacent sources.

Unlike tabular DQ information, the data producer should refrain from creating "nested" DQ coverages at multiple levels, as these can greatly complicate the interpretation of the information. Rather, the user is encouraged to adopt a more integrated approach, where general information is carried on lower level polygons in addition to level-specific information.

40.4.1 Coverage components. VPF provides a variety of mechanisms for storing DQ information within coverages, including attribute tables, standard data quality tables, and optional user-defined relational tables. The mechanisms employed vary depending primarily upon the types of information being stored, rather than the VPF level at which it will reside. DQ coverage attribute tables offer the highest degree of flexibility in storing DQ information, since users can design their own table formats and specifically code those components that vary spatially across the data set. Standard DQ tables (table 46) are particularly useful for data sets where characteristics change radically (with respect to source) from one area to another. In this application, data quality tables are stored as multiple records, with each complete DQ table record corresponding to a face. Within this context, it may be particularly appropriate for data producers to implement subsets or supersets of the standard DQ table. Additional relational tables are useful in normalizing complex data, a technique that is particularly helpful when addressing thematically based variations in DQ information (section 40.4.2.2 of this appendix). Finally, text information is useful for describing phenomena without well-defined spatial extents or for

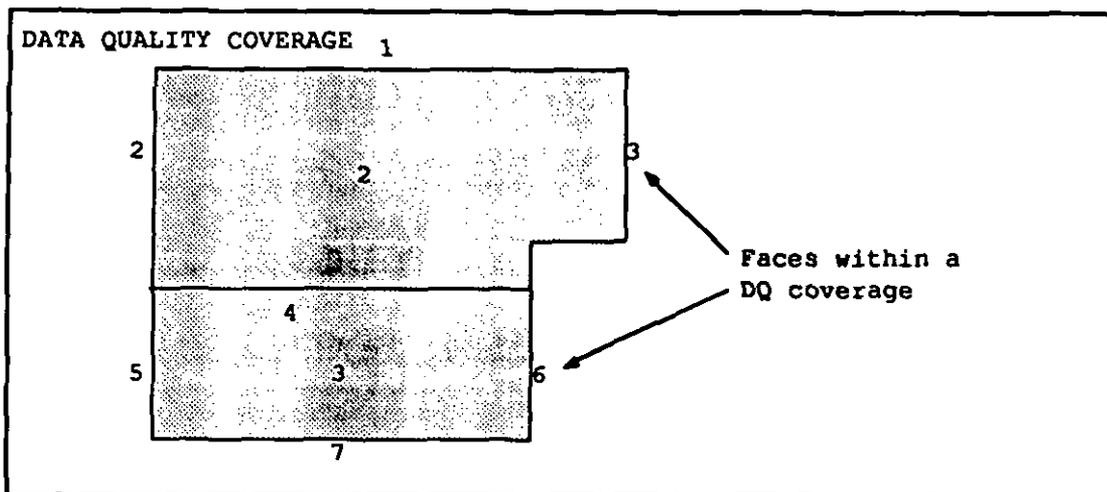
APPENDIX E

annotating special conditions that do not occur with sufficient frequency to justify creating attribute fields to describe them.

40.4.2 Coverage examples. The following sections provide examples of how DQ coverages may be designed under a variety of conditions. The user is encouraged to adopt these approaches when appropriate and to modify them when necessary. Whenever possible, the producer should use coverage level descriptive tables to document the strategy employed in designing and developing DQ coverages for a database.

40.4.2.1 Shared regions and common attributes. The most simple DQ coverage is one where spatial variation of DQ information is shared by all coverages within a library, and DQ attributes are well defined. Recognizing that some nonstandard DQ information may be associated with certain regions, simple relational tables with fields keyed to coverage identifiers can be constructed (figure 53). Edge attributes may also be stored in a separate table.

APPENDIX E



DQ.AFT			
Id	Date	Source	Reliability
1	Null	Null	Null
2	6/78	ONC	Good
3	7/83	ONC	Fair

DQ.LFT	
Id	Comments
1	None
2	Some features badly match
3	None
4	All features mismatched
5	None
6	Many features mismatched
7	Many features mismatched

DQ.COM			
Id	DQ.AFT ID	Coverage	Comments
1	Null	Null	Null
2	2	HYNET	Some contours are missing

Related table for other coverage comments

FIGURE 53. Data quality coverage design.

40.4.2.2 Coverage-specific information. Section 40.4.2.1 describes the basic constructs for describing characteristics of any data set with common spatial components to the reliability information. However, in some cases, the data producer may wish to describe characteristics within a single coverage, where quality information has spatial extents that vary from coverage to coverage. Figure 54 describes a scenario for organizing information under these conditions. The basic approach is to develop a single integrated coverage where the smallest faces are the product of the intersection of the various coverage-based data. Coverages organized in this manner are relatively easy to maintain, particularly for selective updating where new faces affect more than one primary data coverage. An alternative approach would be to maintain a series of separate coverages.

40.5 Conclusions. VPF provides a number of options for encoding data quality information. The information itself can be encoded at any level within the VPF structure depending upon its

APPENDIX E

basic thematic and spatial characteristics. VPF data producers are encouraged to make use of the data quality table whenever possible. In instances where DQ characteristics vary spatially, the use of data quality coverages is strongly recommended.

APPENDIX E

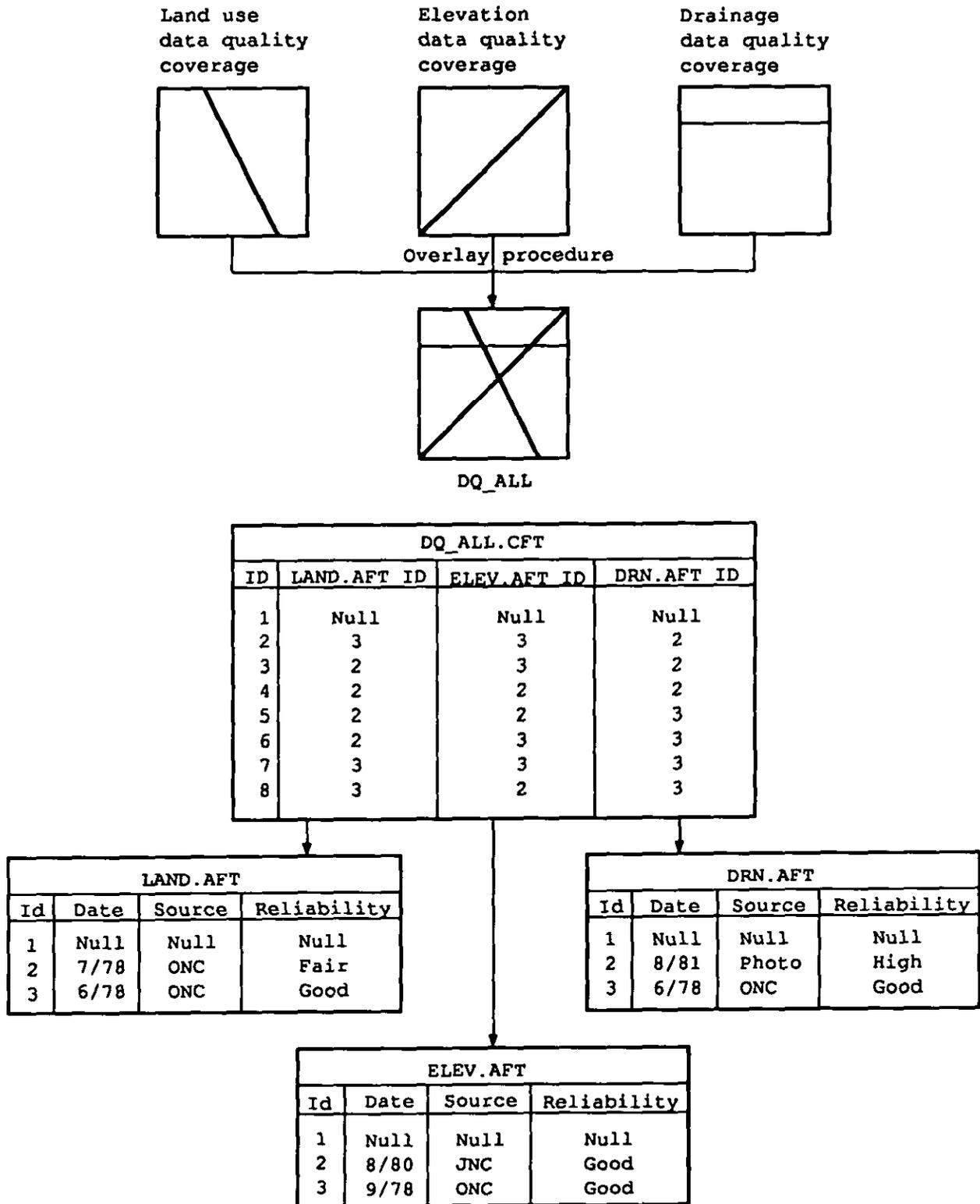


FIGURE 54. Data quality coverage design.

APPENDIX F

SPATIAL INDEXING

10. GENERAL

10.1 Scope. This appendix provides information and discussion concerning spatial indexes in a VPF database. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

20. APPLICABLE DOCUMENTS

This section is not applicable to this appendix.

30. DEFINITIONS

For purposes of this appendix, the definitions of section 3 of the main document shall apply.

40. GENERAL INFORMATION

40.1 Introduction. Spatial queries are queries in which the user points at a specific position on a display device containing a graphic representation of the data and asks (for example) "What is this line?" In order to answer the spatial query, any software that conducts a spatial query on a VPF database must search the edge primitive table for an exact match with "this line." Without a spatial index, the software would have to search every vertex of every edge sequentially for the correct response.

The purpose of a spatial index is to improve the speed with which software can retrieve a specific set of row ids from a primitive table. If the database contains spatial indexes, the software, when given a spatial query like, "What are the features within this bounding region?" can quickly retrieve the primitives that match the query. For each primitive (face, edge, node, and text), there can exist a spatial index file: FSI, ESI, NSI, or TSI (see section 5.4.2).

40.2 Categories of spatial decomposition. The spatial index is the second of four categories of spatial decomposition of a VPF database. The other three are the tile directory, the minimum bounding rectangle of the edge and face primitives, and the primitive coordinates. All four categories of spatial decomposition are described below.

APPENDIX F

40.2.1 Tile directory. Tiles in an implementation of VPF maintain spatially distributed primitives in separate directories. Thus, software developed for a tiled VPF database can search for data in only the relevant tile after the appropriate tile has been identified.

40.2.2 Spatial index. The second step in a typical software query is to use the appropriate index file (if one has been created within the database design). It is recommended that spatial index files associated with the primitives be created for every product implementation of VPF. Spatial indexes are discussed further below.

40.2.3 Minimum bounding rectangle (MBR). VPF requires that face and edge primitives have associated bounding rectangle table files—FBR and EBR. These tables allow the rapid retrieval of the primitives' spatial extent and are used by the software after the spatial index routine generates the primitive ids for the current spatial query. The bounding rectangle coordinates are typically used by the software to check the validity of the primitive ids in satisfying the query.

40.2.4 Primitive coordinates. It is necessary for software to exhaustively check nodes and text primitives for satisfaction of a spatial query, since these primitives do not have associated minimum bounding rectangles. The coordinate of the primitive is thus used to ensure the accurate retrieval of primitive ids output from the spatial index.

40.3 VPF spatial index file. The spatial index file internal structure in VPF is based on an adaptive grid binary tree. This method is powerful because it can handle all types of spatial queries (point, rectangle, and amorphous polygon). The input primitives are broken down into a grid-based binary tree. At each cell (of the tree) there is an 8-bit MBR and a list of the primitive ids that are found at this level of the tree.

The tree is created by storing primitive ids at a cell of the tree. If the cell fills to the buffer limit, then the cell is split into right and left sons of the cell. The primitive ids are then distributed into either son depending on the primitive MBRs. If a primitive MBR falls across both tree cells, the primitive id remains in the parent cell.

When examined spatially, the spatial index divides a tile into subelements (the cells of the tree); figure 55. Each split results in dividing the parent cell into half. The first split is into right and left halves; the next split is into top and bottom halves; then right and left halves again; and so on.

The actual format of the spatial index file consists of the following:

APPENDIX F

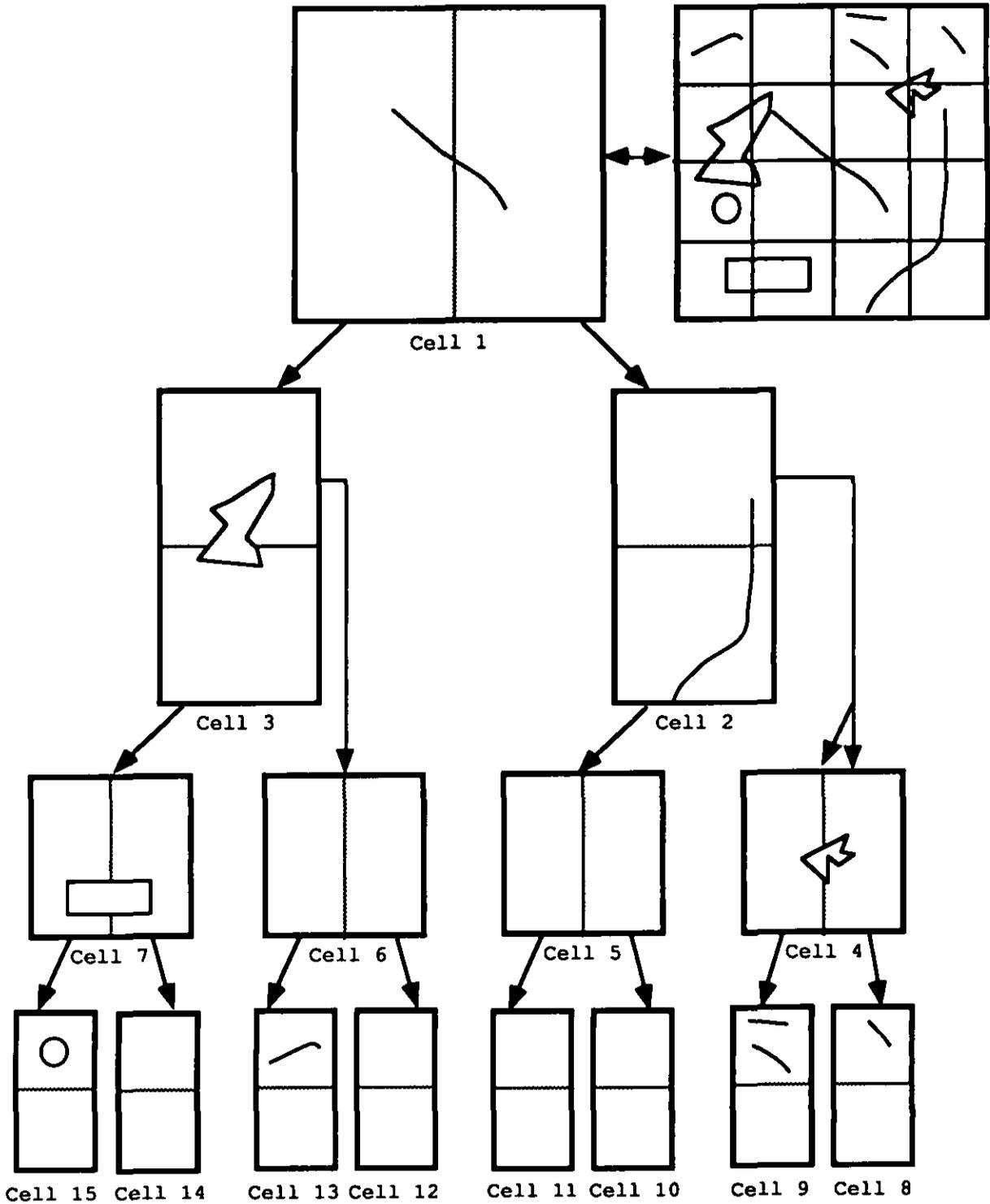


FIGURE 55. Spatial index cell decomposition.

- a. A header containing the number of primitives, the minimum bounding rectangle of the entire spatial extent of the

APPENDIX F

- tree (usually a tile), and the number of cells in the tree.
- b. A bit array of the tree. Each bit represents the tree cell id. It is set to 1 if primitive ids exist at the cell.
 - c. Each tree cell written as a record. The format is an 8-bit integer MBR and a list of primitive ids at that cell of the tree.

40.3.1 Tree navigation. For any cell, the cell from which it was generated is the integer value obtained by dividing by two. Thus, cell 3 points back to cell 1 [INT(3/2)], as does cell 2.

New cells created by splitting are numbered by multiplying the current cell by two and adding one for the second new cell. Thus, cell 2 becomes cells 4 and 5, and cell 3 becomes cells 6 and 7.

40.3.2 Spatial index coordinate system. The coordinate system for the spatial index is based upon 1-byte integers, so a primitive's MBR must be converted to the spatial index coordinate system. All coordinates are relative to the lower left corner of the tile and range from 0 to 255.

40.4 Examples of spatial index creation. Table 62 is a listing of the minimum (x_1 , y_1) and maximum (x_2 , y_2) coordinates of the MBRs of 19 face primitives. The coordinate values in table 62

TABLE 62. Minimum and maximum coordinates for 19 primitives in a tile. Universe or tile boundary is primitive number 1.

Primitive ids	x_1 (deg)	y_1 (deg)	x_2 (deg)	y_2 (deg)
1	-5.00	50.00	0	55.00
2	-5.00	54.63	-3.57	55.00
3	-5.00	52.00	-2.74	55.00
4	-5.00	54.91	-4.99	54.94
5	-5.00	54.76	-4.99	54.77
6	-4.80	54.06	-4.31	54.42
7	-3.28	54.05	-3.17	54.15
8	-0.71	53.54	0	53.74
9	-0.57	53.68	-0.53	53.69
10	-4.60	53.13	-4.05	53.43
11	-4.71	53.24	-4.56	53.33
12	-4.80	52.75	-4.78	52.77
13	-5.00	50.53	-2.35	51.82
14	-4.71	51.63	-4.68	51.65
15	-4.68	51.16	-4.65	51.20
16	-1.03	50.78	-0.95	50.84
17	-1.59	50.58	-1.08	50.77
18	-1.99	50.69	-1.96	50.70
19	-5.00	50.16	-4.99	50.17

APPENDIX F

are all in degrees. The primitives are all located within a 5- by 5-degree tile that has an MBR of (-5, 50), (0, 55). The MBR coordinates can be converted to the spatial index coordinate system by using the following equation:

$$\frac{(\text{Coordinate} - \text{minimum})}{(\text{Maximum} - \text{minimum})} \times 255$$

Our example thus becomes:

$$\frac{(\text{Latitude} - 50)}{(55 - 50)} \times 255$$

and

$$\frac{(\text{Longitude} + 5)}{(0 + 5)} \times 255$$

The results of this conversion are listed in table 63.

TABLE 63. Minimum and maximum spatial index coordinates. Universe or tile boundary is primitive number 1.

Primitive ids	x ₁	y ₁	x ₂	y ₂
1	0	0	255	255
2	0	236	74	255
3	0	102	116	255
4	0	250	1	253
5	0	242	1	244
6	10	206	36	226
7	87	206	94	212
8	218	180	255	191
9	226	187	228	189
10	20	159	49	176
11	14	165	23	170
12	9	140	12	142
13	0	26	136	94
14	14	83	17	85
15	16	59	18	62
16	202	39	207	43
17	173	29	200	40
18	153	35	155	36
19	0	8	1	9

If the MBRs of each primitive are plotted, they appear as shown in figure 56. In figure 57, dividing lines have been added to figure 56 to show that primitive 13 is present in both the left and right halves of the tile, and that primitive 3 is present in both the top and bottom halves of the tile.

APPENDIX F

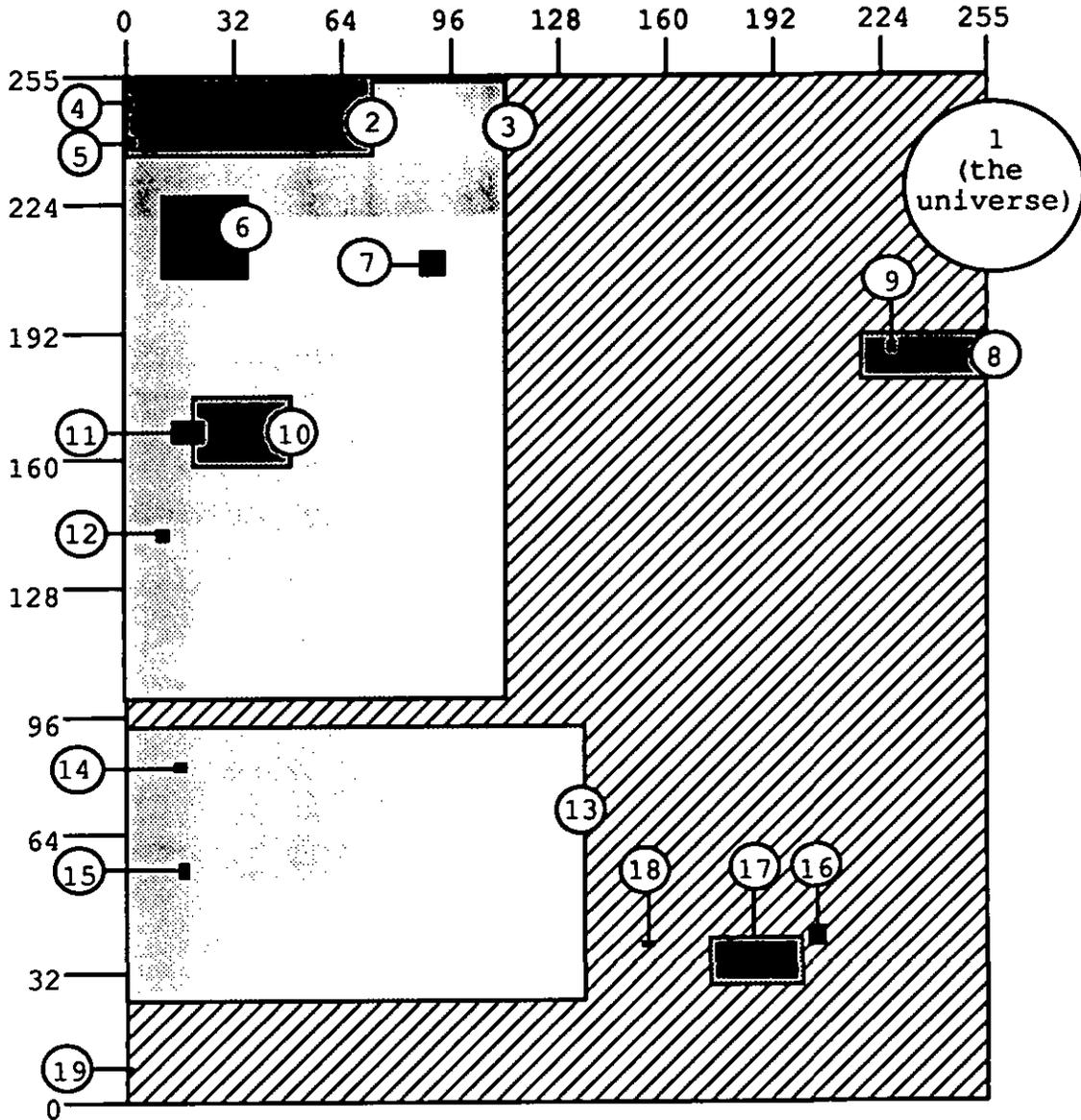


FIGURE 56. Location of MBRs in tile.

APPENDIX F

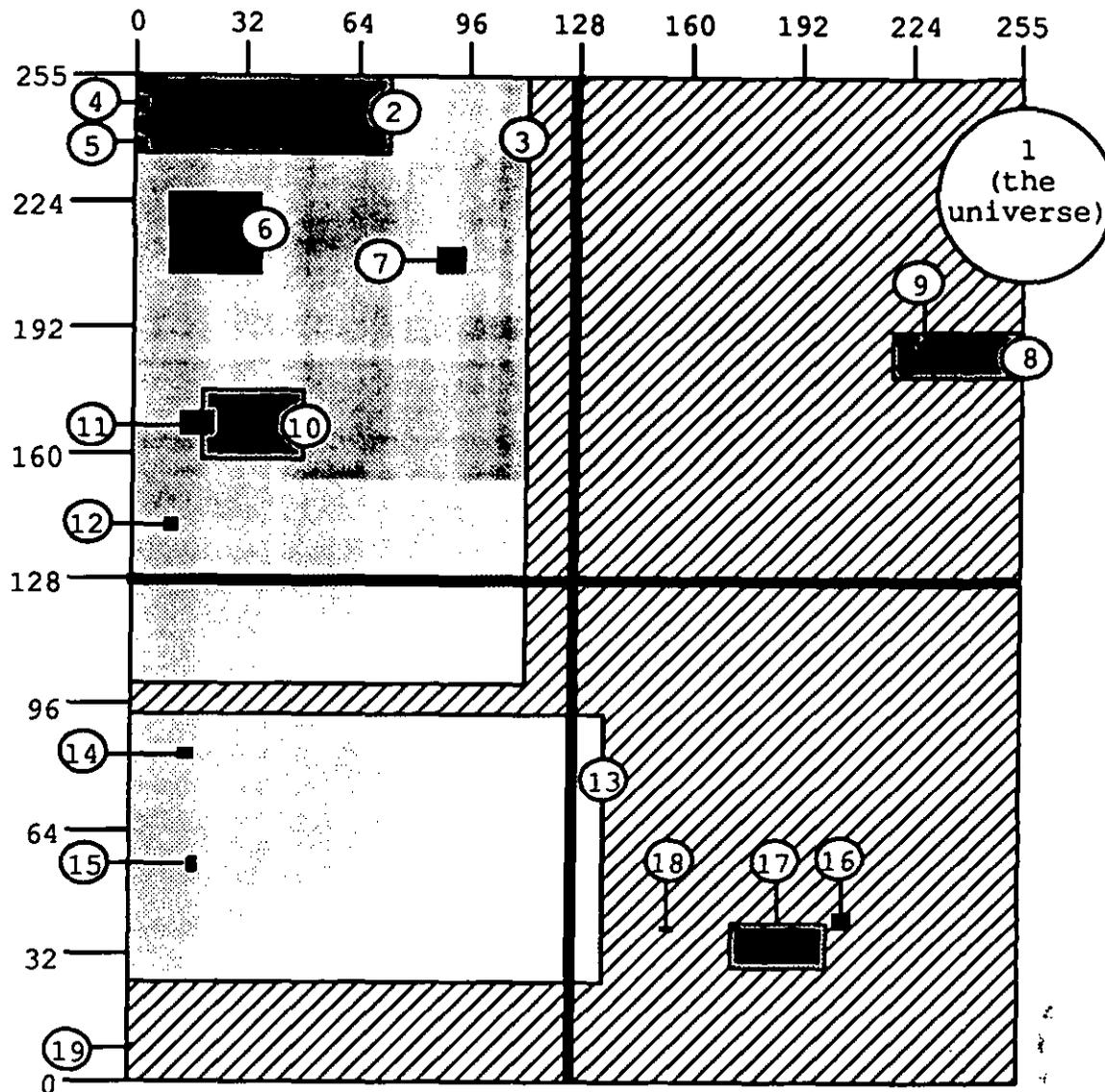


FIGURE 57. Tile content divided in four quarters.

40.4.1 Example of tree creation. The first split of the tile puts primitives 8, 9, 16, 17, and 18 into cell 2 and places primitive 3 into cell 3 (figure 58). Primitive 13 and the universe primitive (1) must be held in cell 1 (figure 59) because neither of the split cells contains the entire MBR for either 13 or 1.

The next split (of cell 3; figure 60) puts primitives 2, 4, 5, 6, 7, 10, 11, and 12 into cell 6 and primitives 14, 15, and 19 into cell 7. Note that no primitives were allocated to cells 4 and 5, since all of the primitives on the right half of the tile could be held in cell 2.

APPENDIX F

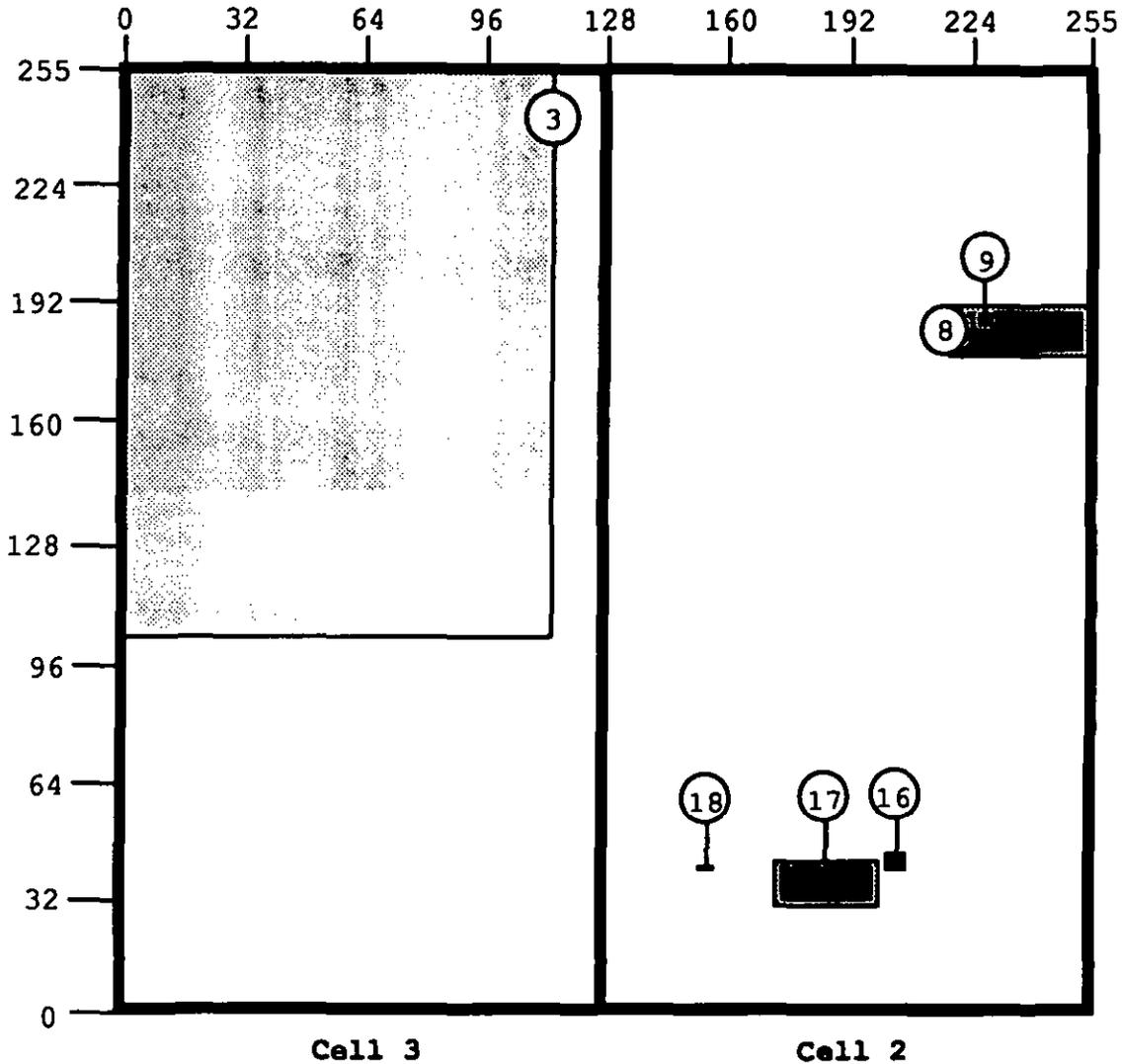


FIGURE 58. First cell split.

All of the primitives in this example have now been allocated to the tree. For more complex tiles, the tree process continues until either (a) no cell has more than some maximum desired number of features; or (b) the subdivision process reaches the coordinate integer limit (255).

The spatial index that results for this example is shown in table 64.

APPENDIX F

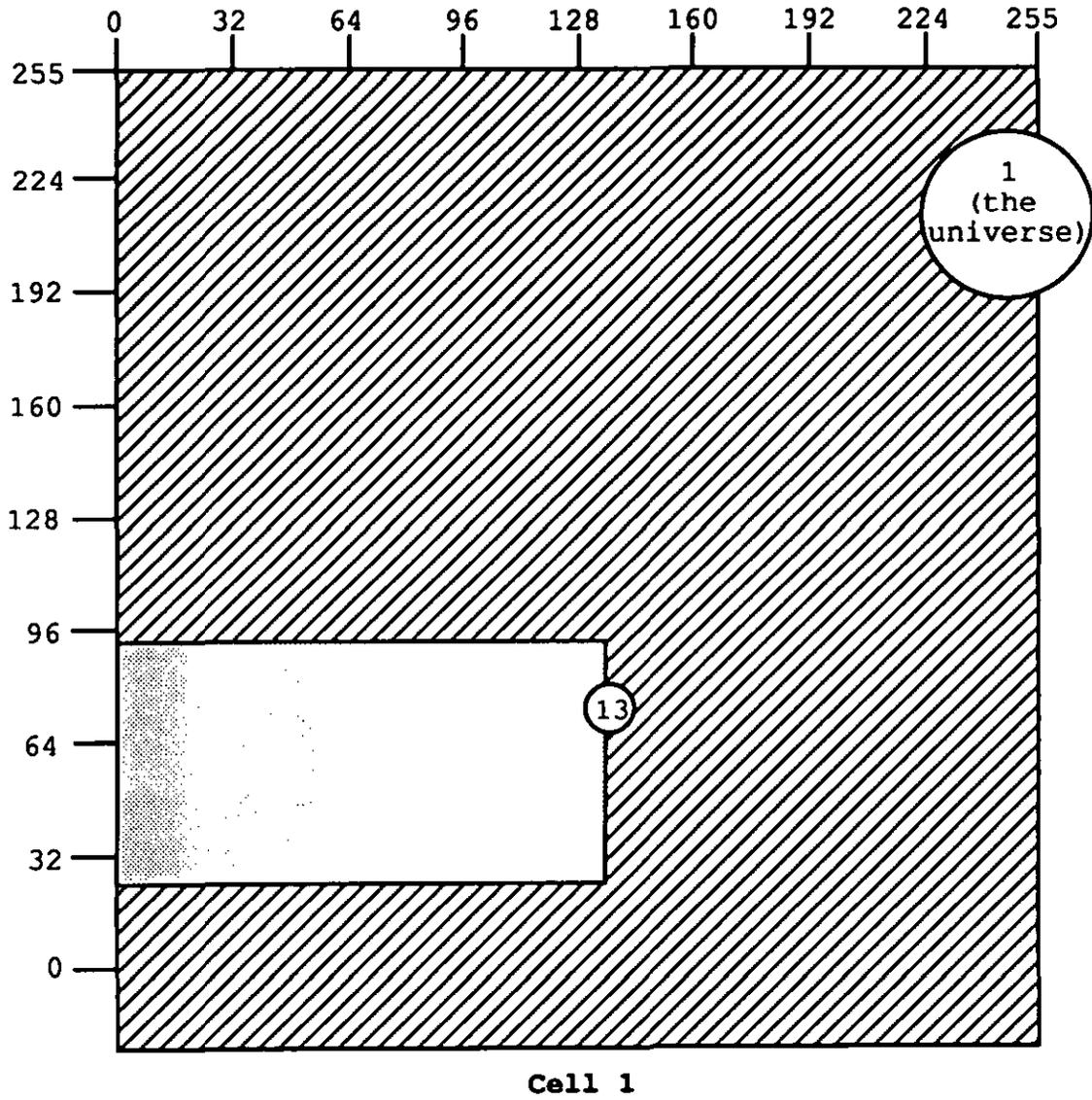


FIGURE 59. Content of cell 1.

APPENDIX F

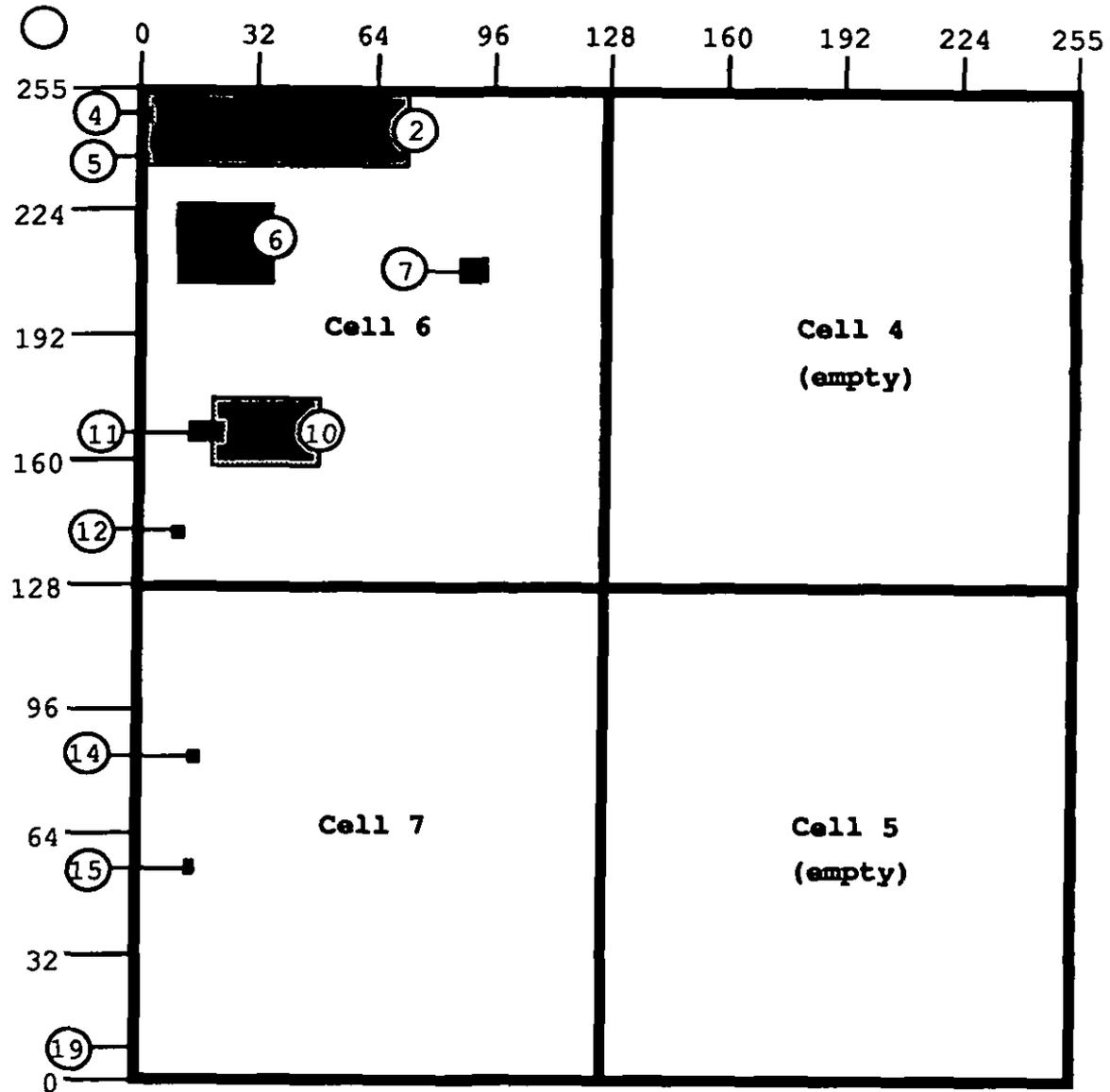


FIGURE 60. Second cell split.

APPENDIX F

TABLE 64. Example of spatial index.

Header:

Number of features: 19
 MBR: -5.00, 50.00, 0.00, 55.00
 Number of cells: 7

Directory:

Cell	Offset	Feature count
1	0	2
2	16	5
3	56	1
6	64	8
7	128	3

Data:

Row	x ₁	y ₁	x ₂	y ₂	Primitive ids
Cell 1					
1	0	26	136	94	13
2	0	0	255	255	1
Cell 2					
3	153	35	155	36	18
4	173	29	200	40	17
5	202	39	207	43	16
6	226	187	228	189	9
7	218	180	255	191	8
Cell 3					
8	0	102	116	255	3
Cell 6					
9	87	206	94	212	7
10	10	206	36	226	6
11	0	242	1	244	5
12	0	250	1	253	4
13	0	236	74	255	2
14	20	159	49	176	10
15	14	165	23	170	11
16	9	140	12	142	12
Cell 7					
17	0	8	1	9	19
18	16	59	18	62	15
19	14	83	17	85	14

40.5 Spatial query. A spatial index can support a spatial query in several ways. For node primitives, software may require the point to be within a specified distance of a pixel designated during the query process, or within a box generated during the query process. For an edge primitive, the software may specify that candidate edges are to be within some specified perpendicular

APPENDIX F

distance of the query pixel or have MBRs that intersect a query box. For a face primitive, the software may define the candidate edges as having MBRs that intersect a query box, be fully contained within the query MBR, or be determined by solving the "point-in-polygon" puzzle. In any case, the spatial index forms the starting point for the database search. It works as follows:

- a. The user designates a query point (pixel).
- b. The software converts the query point into the spatial index coordinate system ((0,0) to (255,255)).
- c. The software determines the smallest cell id.
- d. The software tests all features within the smallest cell, if any, to determine if these features qualify for the query response.
- e. The software calculates the next larger cell (INT of cell id divided by two).
- f. The software repeats the test (if necessary) and continues to increase cell size until cell 1 is reached.

40.6 Spatial query using the sample tree.

- a. The software points at cell 5 and gets nothing.
- b. The software goes to cell 2 and gets faces 8, 9, 16, 17, and 18.
- c. The software checks the MBR of faces found to determine if they include the query point.
- d. The software goes to cell 1 and gets face 13.
- e. The software checks the MBR of face 13 to determine if it includes the query point.
- f. The software reports the query result.

APPENDIX G

CODING FOR METADATA TABLES

The following codes are used in the VPF metadata tables found at the library and database levels. In particular, these coding schemes must be used in the geographic reference table (table 38) to ensure the proper interpretation of the coordinate system in a VPF library.

TABLE 65. Vertical datum codes.

Code	Description
000	Unknown
001	Geodetic
002	High water
003	Higher high water
004	Indian spring low water
005	Low water
006	Lower low water
007	Mean high water
008	Mean high water neaps
009	Mean high water springs
010	Mean higher high water
011	Mean low water
012	Mean low water neaps
013	Mean low water springs
014	Mean lower low water
015	Mean sea level
016	Mean tide level
017	Neap tide
018	Spring tide
019	Mean lower low water springs
020	Lowest astronomical tide
021	Chart datum
022	Highest astronomical tide
023	Geoid
999	Other

TABLE 66. Coding for units of measure.

Code	Description
000	Unknown
001	Meters
003	Seconds
014	Feet
021	Inches
022	Kilometers
999	Other

APPENDIX G

TABLE 67. Coding for ellipsoids.

Code	Description
AAY	Airy
AUN	Australian National
BES	Bessel
CLE	Clarke 1858
CLK	Clarke 1866
CLJ	Clarke 1880
EVE	Everest
FIS	Fischer
GRS	Geodetic Reference System 1980
INT	International
KRA	Krasovsky
AAM	Modified Airy
EVM	Modified Everest
WAL	Walbeck
WGA	World Geodetic System 1960
WGB	World Geodetic System 1966
WGC	World Geodetic System 1972
WGE	World Geodetic System 1984

TABLE 68. Coding for geodetic datums.

Code	Description
AUA	Australian Geodetic
EUR	European 50
ENC	European 87
GEO	Geodetic 1949
NAS	North American 1927
NAX	North American 1983
WGA	World Geodetic System 1960
WGB	World Geodetic System 1966
WGC	World Geodetic System 1972
WGE	World Geodetic System 1984

APPENDIX G

TABLE 69. Coding for projections.

Code	Description
AK	Albers Equal Area
AK	Azimuthal Equal Area
AL	Azimuthal Equal Direct
GN	Gnomonic
LE	Lambert Conformal Conic
LJ	Lambert Equal Area
MC	Mercator
OC	Oblique
OD	Orthographic
PG	Polar Stereographic
TC	Transverse Mercator
UT	UTM

APPENDIX H

SAMPLE VPF DATABASE

10. SCOPE

10.1 Scope. This document provides a sample text description of a VPF database. The information contained in this standard shall be used by the Military Departments, Office of the Secretary of Defense, Organizations of the Joint Chiefs of Staff and the Defense Agencies of the Department of Defense (collectively known as DoD Components) in preparing and accessing digital geographic data required or specified to be in vector product format.

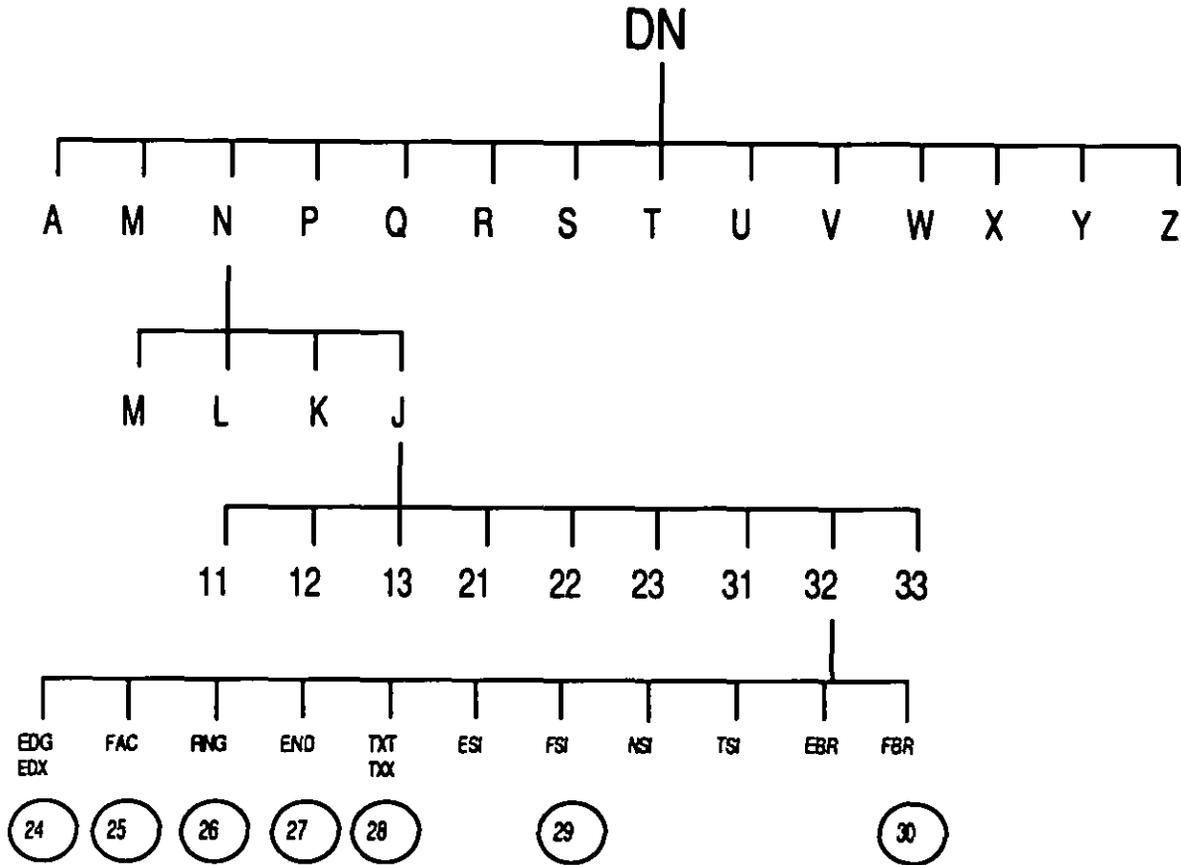
20. EXAMPLES

The following pages document a sample VPF database using the DCW format. The general form includes a printout of the VPF header for each file, followed by the actual data. In some cases the data is truncated or formatted with more than one record per line but in general it can be read by referring to header instructions. For EDG, the actual data was reformatted to show a complete tile layer (without coordinates). This allows cross tile topology for both the right/left edges and right/left faces to be illustrated.

20.1 The DCW data structure contains numerous levels of data with many files at each level. Selected files at various levels are provided herein to illustrate the format and contents. The two figures graphically show the structure and are the roadmap for this document. Files indicated by a TABLE number in a circle by the file identifier correspond to the TABLEs detailed herein. Note that not all files are detailed, only those with the circles.

DCW FILE STRUCTURE

(continued)



APPENDIX H

TABLE 70. Database header table.

The header file...

```

;Database Header Table
;-
;ID=I,                1,P,Row Identifier,-,-,
;DATABASE_NAME=T,     8,N,Directory name of this database,-,-,
;DATABASE_DESC=T,     100,N,Description of this database,-,-,
;MEDIA_STANDARD=T,    20,N,"Media Standard (ie\ : CDROM)",-,-,
;ORIGINATOR=T,        50,N,Producer of this database,-,-,
;ADDRESSEE=T,         100,N,Address of the producer,-,-,
;MEDIA_VOLUMES=T,     1,N,Number of Volumes in this database,-,-,
;SEQ_NUMBERS=T,       1,N,The Sequential Number(s) in this database,-,-,
;NUM_DATA_SETS=T,     1,N,Number of Data Sets,-,-,
;SECURITY_CLASS=T,    1,N,Security Classification,-,-,
;DOWNGRADING=T,       3,N,Downgrading,-,-,
;DOWNGRADE_DATE=D,    1,N,Date,-,-,
;RELEASABILITY=T,     20,N,Releasability restrictions of data,-,-,
;OTHER_STD_NAME=T,    50,N,Description of other data standards used,-,-,
;OTHER_STD_DATE=D,    1,N,Date,-,-,
;OTHER_STD_VER=T,     10,N,Version number of other standard,-,-,
;TRANSMITTAL_ID=T,    1,N,Unique Transmittal Identifier , -,-,
;EDITION_NUMBER=T,    10,N,Edition Number of this database,-,-,
;EDITION_DATE=D,      1,N,Date of edition,-,-,
;
;

```

The actual data...

```

1
DCW
DIGITAL CHART OF THE WORLD: a general purpose global database designed to
support GIS applications
ISO 9660
DEFENSE MAPPING AGENCY/SYSTEM CENTER   HEADQUARTERS, DEFENSE MAPPING AGENCY
ATTN: PR 8613 LEE HIGHWAY, FAIRFAX, VA 22031-2138
4
1
1
1
U
NO 0000000000000000.00000
UNRESTRICTED
DIGEST
1991060000000000.00000
1
1
1
1992030000000000.00000

```

APPENDIX H

TABLE 71. Library attribute table.

The header file...

```
;Library Attribute Table
;-
;ID=I,                1,N,Row Identifier,-,-,
;LIBRARY_NAME=T,      8,P,Library name,-,-,
;XMIN=F,              1,N,Westernmost longitude,-,-,
;YMIN=F,              1,N,Southernmost latitude,-,-,
;XMAX=F,              1,N,Easternmost longitude,-,-,
;YMAX=F,              1,N,Northernmost latitude,-,-,
;
;
```

The actual data...

```
1
NOAMER
-180.00
  5.00
  -5.00
  90.00
2
EURNASIA
-50.00
 35.00
-165.00
 90.00
3
SOAMAFR
-180.00
 -90.00
 180.00
  50.00
4
SAS AUS
 25.00
 -55.00
-130.00
  5.00
5
BROWSE
-180.00
 -90.00
 180.00
  90.00
```

APPENDIX H

TABLE 72. Coverage attribute table.

The header file...

```
;Coverage Attribute Table
```

```
; -
; ID=I,                1,N,Row Identifier,-,-,
; COVERAGE_NAME=T,    8,P,Coverage name,-,-,
; DESCRIPTION=T,      50,N,Coverage description,-,-,
; LEVEL=I,            1,N,Topology level,-,-,
;
;
```

The actual data...

```
1
PO
Political/Oceans
3
2
PP
3
3
Populated Places
LC
Land Cover
3
4
RD
Roads
2
5
RR
2
6
Railroads
2
6
UT
Utilities
2
7
AE
Aeronautical
8
DQ
Data Quality
3
9
```

(table continued)

APPENDIX H

TABLE 72. Coverage attribute table.
(continued)

DN
Drainage
3
10
DS
Supplemental Drainage
11
HY
Hypsography
3
12
HS
Supplemental Hypsography
2
13
CL
Cultural Landmarks
3
14
OF
Ocean Features
2
15
PH
Physiography
2
16
TS
Transportation Structure
2

APPENDIX H

TABLE 73. Geographic reference table.

The header file...

Geographic Reference Table

```
;-
;ID=I,          1,P,Row Identifier,-,-,
;DATA_TYPE=T,  3,N,Data Type,-,-,
;UNITS=T,       3,N,Units,-,-,
;ELLIPSOID=T,   15,N,Ellipsoid,-,-,
;ELLIPSOID_DETAIL=T, 50,N,Ellipsoid Details,-,-,
;VERT_DATUM_REF=T, 15,N,Datum Vertical Reference,-,-,
;VERT_DATUM_CODE=T, 3,N,Vertical Datum Code,-,-,
;SOUND_DATUM=T, 15,N,Sounding Datum,-,-,
;SOUND_DATUM_CODE=T, 3,N,Vertical Datum Code,-,-,
;GEO_DATUM_NAME=T, 15,N,Datum Geodetic Name,-,-,
;GEO_DATUM_CODE=T, 3,N,Datum Geodetic Code,-,-,
;PROJECTION_NAME=T, 20,N,Projection Name,-,-,
;
;
```

The actual data...

```
1
GEO
014
WGS 84
A=6378137,B=6356752 Meters
MEAN SEA LEVEL
015
MEAN SEA LEVEL
015

WGS 84
WGE
DECIMAL DEGREES
```

APPENDIX H

TABLE 74. Library header table.

The header file...

Library Header Table

```
;-
;ID=I,                1,P,Row Identifier,-,-,
;PRODUCT_TYPE=T,     12,N,Product Type,-,-,
;LIBRARY_NAME=T,     12,N,Name,-,-,
;DESCRIPTION=T,      100,N,Description of the library,-,-,
;DATA_STRUCT_CODE=T,  1,N,Data Structure Code,-,-,
;SCALE=I,            1,N,Denominator of the representative fraction,-,-,
;SOURCE_SERIES=T,    15,N,Series,-,-,
;SOURCE_ID=T,        30,N,Identifier of the source reference,-,-,
;SOURCE_EDITION=T,   20,N,Edition number of the source,-,-,
;SOURCE_NAME=T,      100,N,Name of library source,-,-,
;SOURCE_DATE=D,      1,N,Source Date,-,-,
;SECURITY_CLASS=T,   1,N,Security Classification,-,-,
;DOWNGRADING=T,      3,N,Downgrading,-,-,
;DOWNGRADING_DATE=D, 1,N,Date,-,-,
;RELEASABILITY=T,    20,N,Releasability,-,-,
;
;
```

The actual data...

DCW

EURNASIA

This library contains one of the 4 geographic regions of the world from the DCW database

ONC series covering geog. area

VARIES

Operational Navigation Charts or Jet Navigation Charts

19890000000000.00000

U

NO

00000000000000.00000

UN

RESTRICTED

APPENDIX H

TABLE 75. Library data quality table.

The header file...

Library Data Quality Table

```
;LINEAGE.DOC
;ID=I,          1,P,Row Identifier,-,-,
;VPF_LEVEL=T,  8,N,VPF Level,-,-,
;VPF_LEVEL_NAME=T, 20,N,Name of VPF Level,-,-
;FEATURE_COMPLETE=T, *N,Feature Completeness Percent,-,-:ATTRIB_COMPLETE=T,
*,N,Attribute Completeness Percent,-,:LOGICAL_CONSIST=T, *N,Logical
Consistency,-,-
;EDITION_NUM=T,  8,N,Edition Number,-,-
;CREATION_DATE=D, 1,N,Creation Date,-,-,
;REVISION_DATE=D, 1,N,Revision Date,-,-,
;SPEC_NAME=T,    20,N,Product Specification Name,-,-
;SPEC_DATE=D,    1,N,Product Specification Date,-,-:EARLIEST_SOURCE=D,
1,N,Date of Earliest Source,-,-
;LATEST_SOURCE=D, 1,N,Date of Latest Source,-,-
;QUANT_ATT_ACC=T, *N,Standard Deviation of Quantitative Attributes
;QUAL_ATT_ACC=T,  *N,Percent Reliability of Qualitative Attributes
;COLLECTION_SPEC=T, *N,Collection Specification Name,-,-
;SOURCE_FILE_NAME=T, 12,N,Included Source File Name,-,-
;ABS_HORIZ_ACC=T,  *N,Absolute Horizontal Accuracy of VPF Level,-,-
;ABS_HORIZ_UNITS=T, 20,N,Unit of Measure for Absolute Horizontal Accuracy,-,-
;ABS_VERT_ACC=T,   *N,Absolute Vertical Accuracy of VPF Level,-,-
;ABS_VERT_UNITS=T, 20,N,Unit of Measure for Absolute Vertical
Accuracy,-,-,
;REL_HORIZ_ACC=T,  *N,Point to point horizontal accuracy of VPF Level,
;REL_HORIZ_UNITS=T, 20,N,Unit of Measure for Point to Point Horizontal Acc.,-,-
;REL_VERT_ACC=T,   *N,Point to Point Vertical Accuracy of VPF Level,-,-,
;REL_VERT_UNITS=T, 20,N,Unit of Measure for Point to Point Vertical Acc.,-,-
;COMMENTS=T,      *N,Miscellaneous Comments,-,-,
;
```

The actual data

LIBRARY
EURNASIA

100 percent of features depicted on the ONC source materials have been captured.

(table continued)

APPENDIX H

TABLE 75. Library data quality table.
(continued)

100 percent of the features have valid attribute codes assigned to them.6 All data were found to be topologically correct.,No duplicate features are present. All areas are completely described as depicted on the source manuscripts. No undershoots or overshoots are present. All data were consistently captured using the rules described in the narrative associated with this TABLE and in the various feature table narrative files present at the coverage level within the library.

1

19920228000000.00000

19920228000000.00000

DCW

19911207000000.00000

19580401000000.00000

19890101000000.00000

100 percent of attribute codes were reviewed against the source manuscripts.

No formal effort was undertaken to develop a quantitative accuracy statement

100 percent of attribute codes were reviewed against the source manuscripts.

No formal effort was undertaken to develop a qualitative accuracy statement

Operational Navigational Charts (ONC)

N/A +/- 6700 feet:

This figure represents the overall library accuracy. Chart-specific accuracies are available as area feature attributes in the DCW data quality coverage. Detailed horizontal accuracy figures were developed by comparing the positions of well-defined points in the roads, railroads, utility lines, and drainage coverages against sources of higher accuracy, measuring the offsets, and expressing differences as a Circular Map Accuracy figure at a 90% confidence interval. This figure was independently verified by adding known production errors to source chart accuracies in a root sum square calculation. The primary accuracy analysis was performed on one chart only in the prototyping phase of the project.feet

+/- 2000 feet: This figure represents overall library accuracy. Chart-specific accuracies are available as area feature attributes in the DCW data quality coverage. Vertical accuracy figures were developed by comparing elevation contour locations on 1:24,000 scale maps to elevation values at the same location within the digital test. The analysis results are expressed as linear error at a 90% confidence interval.feet

Unknown

N/A

Unknown

N/A

Additional descriptions of data lineage are available in the narrative table associated with this data quality table (called lineage.doc). "

APPENDIX H

TABLE 76. Lineage Document

This table documents the lineage characteristics of the 1:1,000,000-scale component of the Digital Chart of the World (DCW) database. It supplements information in the DCW Data Quality Table (DQT), a standard VPF table residing at the DCW library level that is the main repository of information on source data characteristics. This table contains information on development techniques that are common to all DCW 1:1,000,000-scale library coverages. Information specific to individual feature classes (including special processing techniques, feature coincidence between classes, and database design issues) can be found in narrative files associated with master feature tables at the coverage level of the database. These narrative files are present within individual coverage directories and use a naming convention of the feature class name followed by a .DOC suffix. Quality attributes that vary spatially are stored in the DCW Data Quality (DQ) coverage.

TABLE 77. Gazetteer feature class schema table

The header file...

```
:Gazetteer Feature Class Schema Table
```

```
:ID=I,                1,N,Row Identifier,
:FEATURE_CLASS=T,    8,P,Name of Feature Class,
:TABLE1=T,           12,P,First Table,-,-,
:FOREIGN_KEY=T,      16,P,Column Name in First Table,-,-,
:TASLEZ=T,           12,P,Second Table,-,-,
:PRIMARY_KEY=T,      16,P,Column Name in Second Table,-,-,
:
:
```

The actual data...

```
GAZETTE
GAZETTE. PFT
ID
END
ID
2
GAZETTE
END
ID
GAZETTE. PFT
ID
```

APPENDIX H

TABLE 78. Library reference feature class schema table.

The header file...

```
;Library Reference Feature Class Schema Table
;ID=I,          1,P,Row Identifier,-,-,
;FEATURE_CLASS=T,  8,F,Name of Feature Class,-,-,
;TABLE1=T,      12,F,First Table,-,-,
;FOREIGN_KEY=T,  16,F,Column Name in First Table,-,-,
;TABLE2=T,      12,F,Second Table,-,-,
;PRIMARY_KEY=T,  16,F,Column Name in Second Table,-,-,
;
;
```

The actual data...

```
1
LIBREF
LIBREF.LFT
ID
EDG
ID
2
LIBREF
EDG
ID
LIBREF.LFT
ID
```

APPENDIX H

TABLE 79. Library reference lines

The header file...

;Library Reference Lines

;-

ID=I, 1,P, Row Identifier,-,-,

:

;

The actual data...

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

31

33

34

35

36

37

38

39

40

41

42

43

44

45

APPENDIX H

TABLE 80. Tile reference feature class schema table.

The header file...

```
;Tile Reference Feature Class Schema Table
;-
;ID=I                               1,N,Row Identifier,-,-,
;FEATURE_CLASS=T,                   8,P, Name of Feature Class,-,-,
;TABLE1=T,                           12,F,First Table,-,-,
;FOREIGN_KEY=T                        16,F, Column Name in First Table,-,-,
;TABLE2=T,                           12,F,Second Table,-,-,
;PRIMARY_KEY=T                       16,F,Column Name in Second Table,-,-,
;
;
```

The actual data

```
1
TILeref
TILeref.AFT
ID
FAC
ID
2
TILeref
FAC
ID
TILeref.AFT
ID
```

APPENDIX H

TABLE 81. Tile Reference Areas.

The header file...

```
;Tile Reference Areas
```

```
;-
```

```
;ID=I           1,P,Row Identifier,-,-,
```

```
:TILE_NAME=T    6,N,Tile Name,-,-,
```

```
:
```

```
;
```

The actual data...

1

2

A\M\13

4

A\M\23

5

M\M\23

6

M\M\33

7

N\M\13

.

.

.

APPENDIX H

20.2 The following four tables illustrate the Browse activity.

TABLE 82. Browse coverage attribute table

The header file

```
;Coverage Attribute Table
;-
;ID=I          1,N,Row Identifier,-,-,
;COVERAGE NAME=T  8,P,Coverage name,-,-,
;DESCRIPTION=T  50,N,Coverage Description,-,-,
;LEVEL=I       1,N,Topology level,-,-,
;
;
```

The actual data...

```
1
CO
ONC Compilation Date
3
2
DV
DataVolume
3
3
DN
Drainage
3
4
GR
Geographic Regions
3
5
DA
Hypsographic Data Availability
3
6
IN
ONC Index
3
7
PO
Political/Oceans
3
8
PP
Populated Places
```

APPENDIX H

TABLE 83. Browse geographic reference table.

The header file...

Geographic Reference Table

```
;-
;ID=I,                1,P,Row Identifier,-,-,
;DATA_TYPE=T,        3,N,Data Type,-,-,
;UNITS=T,            3,N,Units,-,-,
;ELLIPSOID=T,       15,N,Ellipsoid,-,-,
;ELLIPSOID_DETAIL=T, 50,N,Ellipsoid Details,-,-,
;VERT_DATUM_REF=T,   15,N,Datum Vertical Reference,-,-,
;VERT_DATUM_CODE=T,  3,N,Vertical Datum Code,-,-,
;SOUND_DATUM=T,     15,N,Sounding Datum,-,-,
;SOUND_DATUM_CODE=T, 3,N,Vertical Datum Code,-,-,
;GEO_DATUM_NAME=T,  15,N,Datum Geodetic Name,-,-,
;GEO_DATUM_CODE=T,  3,N,Datum Geodetic Code,-,-,
;PROJECTION_NAME=T, 20,N,Projection Name,-,-,
;
;
```

The actual data...

```
1
GEO
014
WGS 84
A=6378137,B=6356752 Meters
MEAN SEA LEVEL
015
MEAN SEA LEVEL
015

WGS 84
WGE
ROBINSON
```

APPENDIX H

TABLE 84. Browse library header table

The header file...

```

;Library Header File
;-
;ID=I          1,P,Row Identifier,-,-,
;PRODUCT TYPE=T 12,N,Product Type,-,-,
;LIBRARY_NAME=T 12,N,Name,-,-,
;DESCRIPTION=T  100,N,Description of the library,-,-,
;DATA_STRUCT_CODE=T 1,N,Data Structure Code,-,-,
;SCALE=I       1,N,Denominator of the representative fraction,-,-,
;SOURCE_SERIES=T 15,N,Series,-,-,
;SOURCE_ID=T    30,N,Identifier of the source reference,-,-,
;SOURCE_EDITION=T 20,N,Edition number of the source,-,-,
;SOURCE_NAME=T  100,N,Name of library source,-,-,
;SOURCE_DATE=D  1,N,Source Date,-,-,
;SECURITY_CLASS=T 1,N,Security Classification,-,-,
;DOWNGRADING=T  3,N,Downgrading,-,-,
;DOWNGRADING DATE=D 1,N,Date,-,-,
;RELEASABILITY=T 20,T,Releasability,-,-,
;
;

```

The actual data...

```

1
DCW
BROWSE
The BROWSE library contains data which supports overivew displays at a global
scale
8
31000000
ORIGINAL COMPIL
N/A
ONE
ORIGINAL COMPILATION
19910000000000.00000
U
NO
00000000000000.00000
UNRESTRICTED

```

APPENDIX H

TABLE 85. Browse library data quality table

The header file...

Library Data Quality Table

```

;-
:ID=I,                1,P,Row Identifier,-,-,
:VPF_LEVEL=T,         8,N,VPF Level,-,-,
:VPF_LEVEL_NAME=T,    20,N,Name of VPF Level,-,
:FEATURE_COMPLETE=T,  *,N,Feature Completeness Percent,-,-,
:ATTRIB_COMPLETE=T,   *,N,Attribute Completeness Percent,-,-,
:LOGICAL_CONSIST=T,   *,N,Logical Consistency,
:EDITION_NUM=T,       8,N,Edition Number,-,-,
:CREATION_DATED,      1,N,Creation Date,
:REVISION_DATED,      1,N,Revision Date,
:SPEC_NAME=T,         20,N,Product Specification Name,
:SPEC_DATED,          1,N,Product Specification Date,
:EARLIEST_SOURCE=D,   1,N,Date of Earliest Source,-,-,
:LATEST_SOURCE=D,     1,N,Date of Latest Source,-,-,
:QUANT_ATT_ACC=T,     *,N,Standard Deviation of Quantitative Attributes,
:COAL_ATT_ACC=T,      *,N,Percent Reliability of Qualitative Attributes,
:COLLECTION_SPEC=T,   *,N,Collection Specification Name,-,-,
:SOURCE_FILE_NAME=T,  12,N,Included Source File Name,
:ABS_HORIZ_ACC=T,     *,N,Absolute Horizontal Accuracy of VPF Level,-,-,
:ABS_HORIZ_UNITS=T,   20,N,Unit of Measure for Absolute Horizontal
Accuracy,-,-,
:ABS_VERT_ACC=T,      *,N,Absolute Vertical Accuracy of VPF Level,
:ABS_VERT_UNITS=T,   20,N,Unit of Measure for Absolute Vertical Accuracy,
:REL_HORIZ_ACC=T,     *,N,Point to point horizontal accuracy of VPF
Level,-,-,
:REL_HORIZ_UNITS=T,   20,N,Unit of Measure for Point to Point Horizontal
Acc,-,-,
:REL_VERT_ACC=T,      *,N,Point to Point Vertical Accuracy of VPF Level,
:REL_VERT_UNITS=T,   20,N,Unit of Measure for Point to Point Vertical
Acc,-,-,
:COMMENTS=T,         *,N,Miscellaneous Comments,-,-,

```

(table continued)

APPENDIX H

TABLE 85. Browse library data quality table
(continued)

The actual data...

LIBRARY

BROWSE

Only portions of features from a variety of sources were included in the original

1 compilation.

Valid attribute codes have been assigned to features.

All data were found to be topologically correct. No duplicate features are present.

ent. All areas are completely described as depicted on the source manuscripts.

No undershoots or overshoots are present.

19910115000000.00000

19910115000000.00000

DCW

19911207000000.00000

00000000000000.00000

00000000000000.00000

No formal effort was undertaken to develop a quantitative accuracy statement.

No formal effort was undertaken to develop a qualitative accuracy statement.

N/A

Type

Unknown. No accuracy evaluation was undertaken.

N/A

Unknown

N/A

Unknown

N/A

The coverages within this library were created for reference only. They are not intended to be used for analytical purposes.

APPENDIX H

20.2. The following tables illustrate selected files in the Drainage Data Layer

TABLE 86. Drainage layer area.doc

The header file ...

QA Metadata for the DNAREA feature class

```
;-
;ID=1,      1,P,Feature Table Primary Key,-,-,
;TEXT=T,    SO,N,Text Information,-,-,
;
;
```

The actual data...

This table describes characteristics of data within the DCW DNAREA (drainage areas) feature class. The information presented applies to drainage area features throughout the library. Three subjects are discussed: 1) special automation techniques, 2) feature coincidence, and 3) database design issues. The table does not contain a full description of the data production process. Data lineage information common to all DCW coverages is presented in the file LINEAGE.DOC, which is a narrative file for the Data Quality Table present at the DCW library level of the database. Quality information that varies spatially across the library is stored in the DCW (DQ) data quality coverage.

Special automation techniques

All drainage area features were automated in conjunction with drainage line features from positive source manuscripts containing both feature types. In many cases, non-perennial inland water features were depicted on source manuscripts with dashed line boundaries. These lines were made continuous by manually connecting the dashed lines into solid line features prior to automation. Water bodies were coded using a semiautomated process of scanning water body mask separates, automatically outlining the solid features with vectorization software, building polygon topology, and copying pre-coded labels to the composite drainage coverage. This process was followed by extensive quality control checks to verify that polygon features were labeled correctly. Text

information was removed from the vector data through a combination of symbol trapping and manual editing. All drainage line and area features, and coastlines were processed in a single coverage through cartographic and attribute code quality control steps.

Feature coincidence

All drainage area features intersect drainage lines at single, coordinate coincident points. All shared boundaries between drainage area features, city outlines, and land cover area features have coordinate coincident representations. Drainage area features were treated as the primary source for

APPENDIX H

common boundaries between the various area data types listed above. That is, in cases where drainage areas were in contact with the other area features on the source, the shared boundary between them was always contributed by the drainage area features.

Database design issues

The minimum feature size for all area feature layers within the DCW is 0.12 inches (circumference measure). DN coverage features smaller than this size are retained as point locations in the DS coverage. There are two exceptions to this rule: (1) lake features comprised of more than one line primitive (as when two different streams intersect a lake) are retained in the DN coverage as area features, and (2) glaciers smaller than the minimum size criteria have been eliminated from the database (without representation in the DS coverage).

The boundaries of some non-perennial water bodies are coded as being made up of both shorelines and streams. This occurs when streams were depicted on the source manuscripts as superseding a shoreline. Boundaries between ocean area features in the PO coverage and water bodies in the DN coverage were captured based on differences in tints on the source manuscripts. Under construction reservoirs are captured as are retained in these cases. Due to coding conventions where large inland seas are included as part of PO coverage ocean aggregations, visually anomalous cases of DN water bodies within ocean aggregations do occur occasionally. These are most prevalent in regions where shorelines exhibit large seasonal fluctuations, leaving behind small intermittent water bodies in the dry sea bed (e.g. the Caspian Sea).

TABLE 87. Drainage Layer line.doc

The header file ...

```
QA Metadata for DNLIN feature class
;-
;ID=I,      1,P,Feature Table Primary Key,-,-,
:TEXT=T,    SO,N,Text Information,-,-,
:
;
```

The actual data...

This table describes characteristics of data within the DCW DNLIN (drainage line) feature class. The information presented applies to drainage line features throughout the library. Three subjects are discussed: 1) special automation techniques, 2) feature coincidence, and 3) database design issues. The table does not contain a full description of the data production process. Data lineage information common to all DCW coverages is presented in the file LINEAGE.DOC, which is a narrative file for the Data Quality table present at the DCW library level of the database. Quality information that varies spatially across the library is stored in the DCW (DQ) data quality coverage.

APPENDIX H

Special automation techniques

Intermittent drainage line symbols were solidified into continuous lines by manually connecting the data on the source using drafting ink. Drainage lines were scanned from source separates that always included coastlines and usually included annotation text. Text information was removed from the vector data through a combination of symbol trapping and manual editing. DNLNTYPE attribute values were set by globally assigning the dominant value present on the sheet, and selectively recoding the exceptions. All drainage line and area features, and coastlines were processed in a single coverage through cartographic and attribute code quality control steps.

Feature coincidence

All drainage line features have coordinate coincidence with coastlines and interior shorelines in places where they are in contact. For example, streams entering lakes end at coordinate positions that coincide with one of the coordinates describing the shoreline. All common boundaries between interior shorelines, and populated place and land cover area features have identical coordinate representations in the separate coverages. Drainage boundaries always took precedence over boundaries contributed from other feature classes. Linear dam features are not coincident with inland shorelines. Dam point features are not coincident with associated drainage line features. DS point features are not coincident with associated DN lines. No effort was made to reconcile drainage lines with elevation data, since source materials often showed poor interaction between these layers. When ice cliffs are coincident with glacier outlines on the source maps, the coincident section has identical coordinate representations in the PH (physiography) and DN coverages.

Database design issues

Features were assigned a DNLNTYPE attribute of canal only when they were explicitly labeled as such on the source manuscript. Intermittent streams were coded as ending at the beginning of the first non-perennial symbol segment. In cases where single line features flowed into water bodies, no attempt was made to generate centerlines down the middle of water bodies. Directionality arrows in arid areas (where streams terminate in open country) were not captured. Unsurveyed streams were coded with DNLNTYPE values derived from map contextual information.

TABLE 88. Drainage layer point feature table

The header file ...

```
;QA Metadata for the DNPOINT feature class
;-
;ID=I,1,P,Feature Table Primary Key,-,-,
;TEXT=T,80,N,Text Information,-,-,
;
;
```

The actual data...

APPENDIX H

This table describes characteristics of data within the DCW DNPOINT (drainage point) feature class. The information presented applies to drainage point features throughout the library. Three subjects are discussed: 1) special automation techniques, 2) feature coincidence, and 3) database design issues. The table does not contain a full description of the data production process. Data lineage information common to all DCW coverages is presented in the file LINEAGE.DOC, which is a narrative file for the Data Quality table present at the DCW library level of the database. Quality information that varies spatially across the library is stored in the DCW (DQ) data quality coverage.

Special automation techniques

All drainage points were manually digitized.

Feature coincidence

Drainage point positions do not necessarily coincide with associated drainage line and area feature positions.

Database design issues

All dams are captured as points in this layer. Dams represented by lines longer than 0.25 inch that display character (indicating actual shape of the real world object) are also captured as line features in the CL coverage.

TABLE 89. Drainage Layer area feature table

The header file...			
;Drainage Areas			
;DNAREA.DOC			
;ID=I,		1,P,Row Identifier,-,-,	
:DNPYTYPE=I,		1,N,Type of Drainage Area,INT.VDT,-,	
:TILE_ID=S		1,F,Tile Reference Identifier,-,DNAREA.ATI,	
:FAC_ID=I,		1,F,Face Primitive Foreign Key,-,-,	
:			
;			
The actual data...			
4569	1	410	140
4570	0	457	1
4571	1	457	2
4572	1	457	3
4573	1	457	4

(table continued)

APPENDIX H

TABLE 89. Drainage Layer area feature table
(continued)

4574	-9	457	5
4575	-9	457	6
4576	-9	457	7
4577	-9	457	8
4578	1	457	9
4579	1	457	10
4580	1	457	11
4581	1	457	12
4582	1	457	13
4583	1	457	14
4584	1	457	IS
4585	-9	457	16
4586	-9	457	17
4587	-9	457	18
4588	-9	457	19
4589	-9	457	20
4590	-9	457	21
4591	-9	457	22
4592	-9	457	23
4593	-9	457	24
4594	-9	457	25
4595	-9	457	26
4596	-9	457	27
4597	-9	457	28
4598	-9	457	29
4599	-9	457	30
4600	-9	457	31
4601	-9	457	32
4602	-9	457	33
4603	-9	457	34
4604	1	457	35
4605	1	457	36
4606	2	457	37
4607	-9	457	38
4608	I	457	39
4609	2	457	40
4610	1	457	41
4611	2	457	42
4612	1	457	43
4613	-9	457	44
4614	-9	457	45
4615	2	457	46
4616	2	457	47
4617	2	457	48
4618	2	457	49
4619	-9	457	50
4620	2	457	51
4621	2	457	52
4622	2	457	53
4623	1	457	54

(table continued)

APPENDIX H

TABLE 89. Drainage Layer area feature table
(continued)

4624	1	457	55
4625	1	457	56
4626	2	457	57
4627	2	457	58
4628	2	457	59
4629	-9	457	60
4630	1	457	61
4631	2	457	62
4632	0	411	1
4633	1	411	2

TABLE 90. Drainage Layer area thematic index table

The Drainage Layer Area Thematic Index (For Eurasia Coverage)

The 60 Characters of Header Information...

The size of the HEADER is 3110 bytes
The number of entries = 305
The number of rows = 114808
'T' for Thematic 'G' for Gazetteer... T
Char type of data element being indexed... s
No. of data elements comprising one directory entry... 1
Char data type specifier for the data (S or I)... I
Name of the VPF table from which the index is derived... DNAREA.AFT
Name of column in the VPF table from which index is pulled... TILE_ID°

The Directory follows (Director Id Number has been added)...

1.	255	3110	2	2.	300	3118	17	3.	256	3186	16
4.	210	3250	3	5.	166	3262	370	6.	125	4742	2
7.	211	4750	110	8.	167	5190	121	9.	212	5674	286
10.	168	6818	132	11.	448	7346	2	12.	449	7354	6
13.	213	7378	137	14.	169	7926	167	15.	214	8594	263
16.	170	9646	160	17.	453	10286	161	18.	407	10930	109
19.	454	11366	112	20.	408	11814	249	21.	308	12810	17
22.	309	12878	410	23.	260	14518	273	24.	359	15610	188
25.	310	16362	368	26.	261	17834	321	27.	215	19118	13
28.	171	19170	30	29.	216	19290	9	30.	128	19326	6
31.	217	19350	14	32.	455	19406	48	33.	409	19598	226
34.	456	20502	81	35.	410	20826	140	36.	457	21386	62
37.	411	21634	176	38.	360	22338	280	39.	311	23458	195
40.	361	24238	441	41.	312	26002	345	42.	263	27382	771
43.	362	30466	487	44.	313	32414	913	45.	264	36066	1143
46.	218	40638	7	47.	219	40666	905	48.	220	44286	1113
49.	176	48738	201	50.	91	49542	80	51.	458	49862	35
52.	412	50002	143	53.	459	50574	207	54.	413	51402	255

(table continued)

APPENDIX H

TABLE 90. Drainage Layer area thematic index table
(continued)

55.	460	52422	97	56.	414	52810	309	57.	363	54046	381
58.	314	55570	1059	59.	265	59806	910	60.	364	63446	236
61.	315	64390	673	62.	266	67082	197	63.	365	67870	637
64.	316	70418	408	65.	267	72050	938	66.	221	75802	1262
67.	177	80850	1093	68.	133	85222	12	69.	222	85270	680
70.	178	87990	943	71.	134	91762	82	72.	223	92090	2185
73.	179	100830	1049	74.	135	105026	153	75.	92	105638	126
76.	51	106142	6	77.	93	106166	61	78.	52	106410	4
79.	94	106426	3	80.	53	106438	3	81.	461	106450	180
82.	415	107170	34	83.	462	107306	185	84.	416	108046	65
85.	463	108306	235	86.	417	109246	374	87.	366	110742	427
88.	317	112450	230	89.	268	113370	589	90.	367	115726	362
91.	318	117174	66	92.	269	117438	286	93.	368	118582	593
94.	319	120954	170	95.	270	121634	125	96.	224	122134	1602
97.	180	128542	1139	98.	136	133098	3	99.	225	133110	716
100.	181	135974	495	101.	226	137954	433	102.	182	139686	312
103.	55	140934	2	104.	464	140942	363	105.	418	142394	468
106.	465	144266	267	107.	419	145334	287	108.	466	146482	467
109.	420	148350	781	110.	369	151474	1971	111.	320	159358	490
112.	271	161318	113	113.	370	161770	1170	114.	321	166450	249
115.	272	167446	89	116.	371	167802	494	117.	322	169778	107
118.	273	170206	92	119.	227	170574	299	120.	183	171770	239
121.	228	172726	275	122.	184	173826	502	123.	140	175834	97
124.	229	176222	132	125.	185	176750	455	126.	141	178570	84
127.	57	178906	7	128.	99	178934	2	129.	58	178942	16
130.	100	179006	15	131.	59	179066	37	132.	467	179214	361
133.	421	180658	642	134.	468	183226	710	135.	422	186066	652
136.	469	188674	1311	137.	423	193918	544	138.	372	196094	1583
139.	323	202426	896	140.	274	206010	576	141.	373	208314	892
142.	324	211882	584	143.	275	214218	1292	144.	374	219386	1000
145.	325	223386	814	146.	276	226642	935	147.	230	230382	312
148.	186	231630	397	149.	142	233218	1	150.	231	233222	1422
151.	187	238910	951	152.	143	242714	167	153.	232	243382	2315
154.	188	252642	933	155.	144	256374	275	156.	101	257474	19
157.	60	257550	11	158.	102	257594	3	159.	61	257606	3
160.	470	257618	398	161.	424	259210	502	162.	425	261218	585
163.	426	263558	404	164.	375	265174	494	165.	326	267150	1592
166.	277	273518	676	167.	376	276222	204	168.	327	277038	607
169.	278	279466	526	170.	377	281570	281	171.	328	282694	184
172.	279	283430	236	173.	233	284374	1434	174.	189	290110	791
175.	145	293274	424	176.	234	294970	362	177.	190	296418	839
178.	146	299774	218	179.	235	300646	357	180.	191	302074	718
181.	147	304946	514	182.	63	307002	2	183.	105	307010	5
184.	64	307030	2	185.	106	307038	4	186.	427	307054	171
187.	428	307738	174	188.	429	308434	226	189.	378	309338	287
190.	329	310486	340	191.	280	311846	112	192.	379	312294	223
193.	330	313186	311	194.	281	314430	73	195.	380	314722	302
196.	331	315930	130	197.	282	316450	103	198.	236	316862	57
199.	192	317090	184	200.	148	317826	297	201.	237	319014	16
202.	193	319078	104	203.	149	319494	709	204.	238	322330	43
205.	194	322502	361	206.	150	323946	561	207.	107	326190	17

(table continued)

APPENDIX H

TABLE 90. Drainage Layer area thematic index table
(continued)

208.	66	326258	12	209.	108	326306	91	210.	67	326670	6
211.	109	326694	105	212.	430	327114	426	213.	431	328818	389
214.	432	330374	239	215.	381	331330	391	216.	332	332894	290
217.	283	334054	106	218.	382	334478	524	219.	333	336574	352
220.	284	337982	281	221.	383	339106	573	222.	334	341398	142
223.	285	341966	291	224.	239	343130	75	225.	195	343430	87
226.	151	343778	368	227.	240	345250	155	228.	196	345870	87
229.	152	346218	313	230.	241	347470	321	231.	197	348754	61
232.	153	348998	309	233.	110	350234	26	234.	111	350338	76
235.	433	350642	503	236.	434	352654	169	237.	435	353330	86
238.	384	353674	864	239.	335	357130	66	240.	286	357394	67
241.	385	357662	333	242.	336	358994	215	243.	287	359854	50
244.	386	360054	444	245.	337	361830	107	246.	288	362258	139
247.	242	362814	1099	248.	198	367210	1322	249.	154	372498	340
250.	243	373858	880	251.	199	377378	306	252.	155	378602	1082
253.	244	382930	505	254.	200	384950	132	255.	156	385478	242
256.	436	386446	9	257.	437	386482	108	258.	438	386914	12
259.	387	386962	234	260.	338	387898	320	261.	289	389178	249
262.	388	390174	72	263.	339	390462	158	264.	290	391094	112
265.	389	391542	6	266.	291	391566	46	267.	245	391750	258
268.	201	392782	270	269.	157	393862	689	270.	246	396618	461
271.	202	398462	786	272.	158	401606	993	273.	247	405578	241
274.	203	406542	1415	275.	159	412202	1055	276.	116	416422	4
277.	117	416438	79	278.	118	416754	7	279.	390	416782	7
280.	292	416810	48	281.	342	417002	116	282.	293	417466	167
283.	343	418134	4	284.	294	418150	411	285.	248	419794	313
286.	204	421046	2301	287.	160	430250	705	288.	249	433070	162
289.	205	433718	1959	290.	161	441554	385	291.	250	443094	390
292.	206	444654	900	293.	162	448254	3	294.	344	448266	4
295.	295	448282	5	296.	345	448302	2	297.	346	448310	4
298.	251	448326	753	299.	207	451338	332	300.	252	452666	745
301.	208	455646	497	302.	164	457634	1	303.	253	457638	485
304.	209	459578	682	305.	165	462306	9				

The Indexed Data follows...

TABLE 91. Drainage Layer integer value description table

The header file...

```

;Drainage Integer Value Description table
;-
;ID=I,          1,N,Row Identifier,-,-,
;tableT,       12,P,Name of Feature table,-,-,
;ATTRIB[ITE=T, 16,P,Attribute Name,-,-,
;VALUE=I,      1,P,Attribute Value,-,-,
;DESCRIPTION=T,50,N,Attribute Value Description,-,-,
;

```

(table continued)

APPENDIX H

TABLE 91. Drainage Layer integer value description table
(continued)

;
The actual data...
1
DNAREA.AFT
DNPYTYPE
1
Inland water - perennial
2
DNAREA.AFT
DNPYTYPE
2
Inland water - non-perennial
.3
DNAREA.AFT
DNPYTYPE
.3
Wet sand
4
DNAREA.AFT
DNPYTYPE
4
Snowfields,glaciers, ice
5
DNLNLINE.LFT
DNLNNTYPE
1
Streams,rivers,channelized rivers
6
DNLNLINE.LFT
DNLNNTYPE
2
Inland shorelines
7
DNLNLINE.LFT
DNLNNTYPE
3
Wet sand limits
8
DNLNLINE.LFT
DNLNNTYPE
.4
Canals, aqueducts, flumes, penstocks, kanats
9
DNLNLINE.LFT
DNLNNTYPE
5
Glacial limits

(table continued)

APPENDIX H

TABLE 91. Drainage Layer integer value description table
(continued)

10
DNLINE.LFT
DNLNTYPE
6
Snowfield,glacier,land ice to water ice or ocean
11
DNLINE.LFT
DNLNTYPE
7
Ice free limits (land/ice line)
12
DNLINE.LFT
DNLNTYPE
8
Connectors
13
DNLINE.LFT
DNLNSTAT
1
Perennial (used for streams only)
14
DNLINE.LFT
DNLNSTAT
2
Non-perennial (used for streams only)
15
DNLINE.LFT
DNLNSTAT
3
Definite (used for inland shorelines only)
16
DNLINE.LFT
DNLNSTAT
4
Indefinite (used for inland shorelines only)
17
DNLINE.LFT
DNLNSTAT
5
Unsurveyed perennial (used for streams only)
18
DNLINE.LFT
DNLNSTAT
6
Unsurveyed non-perennial (used for streams only)
19
DNLINE.LFT
DNLNSTAT
7
Abandoned

(table continued)

APPENDIX H

TABLE 91. Drainage Layer integer value description table
(continued)

20
DNLINE.LFT
DNLNSTAT
8
Under construction
21
DNLINE.LFT
DNLNSTAT
9
Suspended or elevated
22
DNLINE.LFT
DNLNSTAT
10
Underground
23
DNLINE.LFT
DNLNSTAT
11
Above ground
24
DNLINE.LFT
DNLNSTAT
88
ONC modular boundary
25
DNPOINT.PFT
DNPTTYPE
1
Springs, wells, or waterholes
26
DNPOINT.PFT
DNPTTYPE
2
Reservoir
27
DNPOINT.PFT
DNPTTYPE
3
Falls
28
DNPOINT.PFT
DNPTTYPE
4
Rapids
29
DNPOINT.PFT
DNPTTYPE
5
Locks

(table continued)

APPENDIX H

TABLE 91. Drainage Layer integer value description table
(continued)

30
DNPOINT.PFT
DNPTTYPE
6
Dams
31
DNPOINT.PFT
DNPTTYPE
7
Sluice Gate
32
DNTEXT.TFT
LEVEL
1
Running streams
33
DNTEXT.TFT
LEVEL
2
Standing lakes and reservoirs
34
DNTEXT.TFT
LEVEL
3
Land ice names
35
DNTEXT.TFT
LEVEL
4
Running streams (diacritical)
36
DNTEXT.TFT
LEVEL
5
Standing lakes and reservoirs (diacritical)
37
DNTEXT.TFT
LEVEL
6
Land ice names (diacritical)
38
DNTEXT.TFT
LEVEL
7
DNPOINT regular annotation
39
DNTEXT.TFT
LEVEL
8
DNPOINT diacritical annotation
40

(table continued)

APPENDIX H

TABLE 91. Drainage Layer integer value description table
(continued)

```

DNTEXT.TFT
SYMBOL
1
Black annotation
41
DNTEXT.TFT
SYMBOL
4
Blue annotation

```

TABLE 92. Drainage Layer feature class schema table

```

The header file...

;Drainage Feature Class Schema Table
;-
;ID=I                      1,N,Row Identifier,-,-,
;FEATURE_CLASS=T,         8,P,Name of Feature Class,-,-,
;table1=T,                12,P,First Table,-,-,
;FOREIGN_KEY=T,          16,P,Column Name in First Table,-,-,
;table2=T,                12,P,Second Table,-,-,
;
;

The actual data...

1
DNAREA
DNAREA.AFT
FAC_ID
FAC
ID
2
DNAREA
FAC
DNAREA.AFT_ID
DNAREA.AFT
ID
3
DNLN
DNLN.LFT
EDG_ID
EDG
ID
4
DNLN

```

(table continued)

APPENDIX H

TABLE 92. Drainage Layer feature class schema table
(continued)

EDG
DNLINE.LFT_ID
DNLINE.LFT
ID
5
DNPOINT
DNPOINT.PFT
END_ID
END
ID
6
DNPOINT
END
DNPOINT.PFT_ID
DNPOINT.PFT
ID
7
DNTEXT
DNTEXT.TFT
TXT_ID
TXT
ID
TXT
DNTEXT.TFT_ID
DNTEXT.TFT
ID

20.3 The following tables illustrate the tile data. Tile NJ32 is used as an example.

TABLE 93. Edge Records for NJ32

ID	LINE ID	SN	EN	Rt Face d T1 id	Lt Face ld T1 id	Rt Edge ld T1 id	Lt Edge ld T1 id	No. Coord
1.	24176	109	110	8 - -	6 - -	104 - -	99 - -	2
2.	24177	249	217	14 - -	16 - -	205 - -	3 - -	11
3.	24178	249	241	16 - -	17 - -	231 - -	4 - -	3
4.	24179	249	250	17 - -	14 - -	6 - -	2 - -	2
5.	24180	251	250	14 - -	20 - -	4 - -	9 - -	2
6.	24181	257	250	20 - -	17 - -	5 - -	7 - -	2
7.	24182	257	259	17 - -	22 - -	8 - -	10 - -	3
8.	24183	259	242	17 - -	18 - -	232 - -	12 - -	2
9.	24184	251	260	20 - -	21 - -	247 - -	240 - -	3
10.	24185	257	263	22 - -	20 - -	11 - -	6 - -	2
11.	24186	263	267	22 - -	24 - -	12 - -	257 - -	2
12.	24187	267	259	22 - -	18 - -	7 - -	13 - -	2

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Lt Face Id T1 id	Rt Edge Id T1 id	Lt Edge Id T1 id	No. Coord
13. 24188	267	269	18 - -	25 - -	14 - -	16 - -	2
14. 24189	245	269	19 - -	18 - -	15 - -	233 - -	6
15. 24190	269	276	19 - -	25 - -	259 - -	13 - -	4
16. 24191	279	267	24 - -	25 - -	11 - -	264 - -	3
17. 24192	296	293	26 - -	27 - -	275 - -	283 - -	2
18. 24193	394	393	33 - -	34 - -	366 - -	370 - -	2
19. 24194	477	466	45 - -	44 - -	440 - -	437 - -	20
20. 24195	562	557	49 - -	49 - -	20 - -	510 - -	2
21. 24196	609	609	54 - -	53 - -	21 - -	21 - -	11
22. 24197	611	611	56 - -	53 - -	22 - -	22 - -	7
23. 24198	612	612	55 - -	53 - -	23 - -	23 - -	14
24. 24199	1	7	1 - -	1 - -	24 - -	24 - -	11
25. 24200	4	8	1 - -	1 - -	39 - -	25 - -	5
26. 24201	10	12	1 - -	1 - -	26 - -	26 - -	8
27. 24202	18	16	1 - -	1 - -	27 - -	27 - -	8
28. 24203	11	23	1 - -	1 - -	28 - -	28 - -	15
29. 24204	5	25	1 - -	1 - -	29 - -	29 - -	19
30. 24205	3	26	1 - -	1 - -	30 - -	30 - -	16
31. 24206	2	27	1 - -	1 - -	31 - -	31 - -	22
32. 24207	30	8	1 - -	1 - -	25 - -	32 - -	16
33. 24208	21	33	1 - -	1 - -	33 - -	33 - -	7
34. 24209	20	34	1 - -	1 - -	34 - -	34 - -	3
35. 24210	6	39	1 - -	1 - -	57 - -	35 - -	13
36. 24211	43	36	1 - -	2 - -	37 - -	37 - -	11
37. 24212	36	43	1 - -	2 - -	82 - -	36 - -	5
38. 24213	9	45	1 - -	1 - -	38 - -	38 - -	15
39. 24214	8	46	1 - -	1 - -	39 - -	32 - -	15
40. 24215	49	47	1 - -	1 - -	40 - -	53 - -	2
41. 24216	50	37	1 - -	1 - -	41 - -	41 458 279	5
42. 24217	14	44	1 - -	1 - -	42 - -	42 - -	31
43. 24218	51	40	1 - -	1 - -	43 - -	44 - -	8
44. 24219	51	29	1 - -	1 - -	44 - -	52 - -	8
45. 24220	41	52	1 - -	1 - -	45 - -	45 - -	15
46. 24221	22	55	1 - -	1 - -	64 - -	46 - -	12
47. 24222	56	13	1 - -	1 - -	47 - -	47 - -	20
48. 24223	17	58	1 - -	1 - -	48 - -	48 - -	10
49. 24224	15	59	1 - -	1 - -	49 - -	49 - -	18
50. 24225	60	19	1 - -	1 - -	50 - -	70 - -	13
51. 24226	60	53	1 - -	1 - -	51 - -	50 - -	12
52. 24227	51	65	1 - -	1 - -	61 - -	43 - -	9
53. 24228	49	66	1 - -	1 - -	54 - -	40 - -	9
54. 24229	65	66	1 - -	1 - -	58 - -	52 - -	8
55. 24230	68	61	1 - -	1 - -	55 458 280	56 - -	5
56. 24231	68	31	1 - -	1 - -	56 - -	60 - -	11
57. 24232	69	39	1 - -	1 - -	35 - -	57 - -	7
58. 24233	66	62	1 - -	1 - -	58 - -	53 - -	8
59. 24234	32	72	1 - -	1 - -	119 - -	59 - -	9
60. 24235	68	38	1 - -	1 - -	60 - -	55 - -	17
61. 24236	73	65	1 - -	1 - -	54 - -	62 - -	4

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Lt Face Id T1 id	Rt Edge Id T1 id	Lt Edge Id T1 id	No. Coord
62. 24237	54	73	1 - -	1 - -	87 - -	62 - -	12
63. 24238	24	74	1 - -	1 - -	63 - -	63 - -	13
64. 24239	76	55	1 - -	1 - -	108 - -	64 - -	14
65. 24240	48	77	1 - -	1 - -	66 - -	65 - -	9
66. 24241	77	57	1 - -	1 - -	66 - -	72 - -	13
67. 24242	78	63	1 - -	1 - -	67 - -	76 - -	8
68. 24243	28	79	1 - -	1 - -	73 - -	68 - -	19
69. 24244	83	67	1 - -	1 - -	69 - -	74 - -	5
70. 24245	84	60	1 - -	1 - -	51 - -	79 - -	10
71. 24246	85	42	1 - -	1 - -	71 - -	72 - -	18
72. 24247	85	77	1 - -	1 - -	65 - -	91 - -	2
73. 24248	87	79	1 - -	1 - -	94 - -	77 - -	5
74. 24249	83	88	1 - -	1 - -	79 - -	69 - -	2
75. 24250	89	78	1 - -	3 - -	67 - -	76 - -	7
76. 24251	78	89	1 - -	3 - -	88 - -	75 - -	8
77. 24252	87	90	1 - -	4 - -	102 - -	78 - -	3
78. 24253	91	87	1 - -	4 - -	73 - -	109 - -	2
79. 24254	84	88	1 - -	1 - -	74 - -	86 - -	7
80. 24255	93	86	1 - -	1 - -	80 - -	113 - -	21
81. 24256	72	81	1 - -	1 - -	81 - -	59 - -	17
82. 24257	95	43	1 - -	1 - -	36 - -	83 - -	41
83. 24258	95	64	1 - -	1 - -	83 - -	116 - -	8
84. 24259	96	70	1 - -	1 - -	84 - -	85 - -	5
85. 24260	96	71	1 - -	1 - -	85 - -	138 - -	7
86. 24261	84	98	1 - -	1 - -	86 - -	70 - -	7
87. 24262	73	82	1 - -	1 - -	87 - -	61 - -	17
88. 24263	99	89	1 - -	1 - -	75 - -	93 - -	8
89. 24264	100	99	1 - -	5 - -	88 - -	93 - -	3
90. 24265	101	75	1 - -	1 - -	90 - -	97 - -	7
91. 24266	85	102	1 - -	1 - -	92 - -	71 - -	4
92. 24267	97	102	1 - -	1 - -	120 - -	103 - -	9
93. 24268	99	100	1 - -	5 - -	95 - -	89 - -	4
94. 24269	79	103	1 - -	1 - -	94 - -	68 - -	7
95. 24270	100	104	1 - -	1 - -	100 - -	89 - -	2
96. 24271	104	106	7 - -	1 - -	100 - -	95 - -	4
97. 24272	101	108	1 - -	6 - -	98 - -	101 - -	3
98. 24273	106	108	1 - -	1 - -	99 - -	96 - -	2
99. 24274	108	109	1 - -	6 - -	106 - -	97 - -	2
100. 24275	106	104	7 - -	1 - -	96 - -	98 - -	3
101. 24276	110	101	1 - -	6 - -	90 - -	1 - -	2
102. 24277	90	111	1 - -	4 - -	107 - -	77 - -	3
103. 24278	112	97	1 - -	1 - -	105 - -	104 - -	3
104. 24279	112	110	1 - -	8 - -	101 - -	106 - -	2
105. 24280	97	113	1 - -	1 - -	105 - -	92 - -	3
106. 24281	112	109	8 - -	1 - -	1 - -	103 - -	4
107. 24282	116	111	1 - -	1 - -	111 - -	122 - -	2
108. 24283	117	55	1 - -	1 - -	46 - -	108 - -	22
109. 24284	91	120	4 - -	1 - -	111 - -	78 - -	6
110. 24285	121	119	1 - -	1 - -	110 - -	121 - -	2

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Lt Face Id T1 id	Rt Edge Id T1 id	Lt Edge Id T1 id	No. Coord
111.	24286	120	111	4 - -	1 - -	102 - -	8
112.	24287	120	122	1 - -	1 - -	134 - -	2
113.	24288	93	123	1 - -	1 - -	118 - -	12
114.	24289	35	124	1 - -	1 - -	114 - -	24
115.	24290	126	114	1 - -	1 - -	115 - -	11
116.	24291	95	127	1 - -	1 - -	189 - -	6
117.	24292	93	125	1 - -	1 - -	117 - -	18
118.	24293	123	127	1 - -	1 - -	116 - -	6
119.	24294	72	128	1 - -	1 - -	119 - -	8
120.	24295	130	102	1 - -	1 - -	91 - -	7
121.	24296	130	121	1 - -	1 - -	110 - -	4
122.	24297	116	131	1 - -	1 - -	122 - -	15
123.	24298	92	132	1 - -	1 - -	139 - -	9
124.	24299	115	133	1 - -	1 - -	147 - -	5
125.	24300	105	135	1 - -	1 - -	126 - -	10
126.	24301	118	135	1 - -	1 - -	135 - -	13
127.	24302	107	138	1 - -	1 - -	128 - -	11
128.	24303	138	80	1 - -	1 - -	128 - -	18
129.	24304	139	123	1 - -	1 - -	113 - -	8
130.	24305	140	136	1 - -	1 - -	130 - -	2
131.	24306	133	141	1 - -	1 - -	131 - -	3
132.	24307	142	142	1 - -	10 - -	132 - -	5
133.	24308	135	143	1 - -	1 - -	141 - -	7
134.	24309	144	122	1 - -	1 - -	136 - -	8
135.	24310	135	137	1 - -	1 - -	135 - -	10
136.	24311	122	146	1 - -	1 - -	136 - -	9
137.	24312	148	132	1 - -	9 - -	123 - -	5
138.	24313	149	96	1 - -	1 - -	84 - -	13
139.	24314	151	132	9 - -	1 - -	137 - -	10
140.	24315	152	149	1 - -	1 - -	138 - -	2
141.	24316	152	143	1 - -	1 - -	133 - -	7
142.	24317	151	154	1 - -	9 - -	143 - -	2
143.	24318	155	154	9 - -	1 - -	142 - -	2
144.	24319	148	157	9 - -	1 - -	145 - -	2
145.	24320	157	155	9 - -	1 - -	143 - -	2
146.	24321	134	161	1 - -	1 - -	149 - -	8
147.	24322	163	133	1 - -	1 - -	131 - -	7
148.	24323	163	94	1 - -	1 - -	148 - -	17
149.	24324	161	164	1 - -	1 - -	150 - -	2
150.	24325	164	152	1 - -	1 - -	140 - -	10
151.	24326	166	165	11 - -	1 - -	160 - -	3
152.	24327	166	167	1 - -	11 - -	153 - -	2
153.	24328	167	168	1 - -	11 - -	154 - -	2
154.	24329	168	169	1 - -	11 - -	155 - -	2
155.	24330	169	170	1 - -	11 - -	156 - -	2
156.	24331	170	164	1 - -	1 - -	149 - -	2
157.	24332	150	171	1 - -	1 - -	158 - -	4
158.	24333	171	165	1 - -	1 - -	151 - -	10
159.	24334	173	138	1 - -	1 - -	127 - -	12

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID	LINE ID	SN	EN	Rt Face Id T'1 id	Lt Face Id T'1 id	Rt Edge Id T'1 id	Lt Edge Id T'1 id	No. Coord
160.	24335	165	170	11 - -	1 - -	155 - -	158 - -	14
161.	24336	174	162	1 - -	1 - -	161 - -	174 - -	4
162.	24337	176	175	1 - -	12 - -	163 - -	163 - -	2
163.	24338	176	175	12 - -	1 - -	162 - -	162 - -	9
164.	24339	178	177	1 - -	1 - -	164 - -	190 - -	2
165.	24340	180	130	1 - -	1 - -	120 - -	166 - -	15
166.	24341	171	180	1 - -	1 - -	176 - -	157 - -	12
167.	24342	182	155	1 - -	1 - -	145 - -	167 - -	13
168.	24343	184	160	1 - -	1 - -	168 - -	208 - -	8
169.	24344	185	140	1 - -	1 - -	130 - -	216 - -	13
170.	24345	172	184	1 - -	1 - -	168 - -	170 - -	10
171.	24346	186	159	1 - -	1 - -	171 - -	172 - -	11
172.	24347	186	145	1 - -	1 - -	172 - -	202 - -	12
173.	24348	187	187	13 - -	1 - -	173 - -	183 - -	11
174.	24349	174	188	1 - -	1 - -	180 - -	161 - -	2
175.	24350	185	156	1 - -	1 - -	175 - -	169 - -	16
176.	24351	190	180	1 - -	1 - -	165 - -	205 - -	6
177.	24352	191	147	1 - -	1 - -	177 - -	182 - -	20
178.	24353	192	163	1 - -	1 - -	147 - -	179 - -	10
179.	24354	191	192	1 - -	1 - -	223 - -	177 - -	3
180.	24355	188	193	1 - -	1 - -	181 - -	174 - -	3
181.	24356	193	194	1 - -	1 - -	181 - -	184 - -	5
182.	24357	191	179	1 - -	1 - -	182 - -	179 - -	13
183.	24358	197	187	1 - -	1 - -	173 - -	198 - -	3
184.	24359	200	193	1 - -	1 - -	180 - -	191 - -	6
185.	24360	158	202	1 - -	1 - -	186 - -	185 - -	13
186.	24361	202	190	1 - -	14 - -	176 - -	237 - -	7
187.	24362	197	201	1 - -	1 - -	187 - -	183 - -	9
188.	24363	203	129	1 - -	1 - -	188 - -	188 - -	20
189.	24364	204	127	1 - -	1 - -	118 - -	189 - -	36
190.	24365	206	178	1 - -	1 - -	164 - -	199 - -	19
191.	24366	200	208	1 - -	1 - -	201 - -	184 - -	8
192.	24367	181	209	1 - -	1 - -	193 - -	192 - -	9
193.	24368	209	206	1 - -	1 - -	190 - -	203 - -	9
194.	24369	210	183	1 - -	1 - -	194 - -	195 - -	10
195.	24370	210	208	1 - -	1 - -	191 - -	199 - -	8
196.	24371	211	196	1 - -	1 - -	196 - -	197 - -	6
197.	24372	211	199	1 - -	1 - -	197 - -	218 - -	3
198.	24373	197	212	1 - -	1 - -	198 - -	187 - -	6
199.	24374	210	206	1 - -	1 - -	193 - -	194 - -	28
200.	24375	207	213	1 - -	1 - -	220 - -	200 - -	8
201.	24376	213	208	1 - -	1 - -	195 - -	200 - -	9
202.	24377	215	186	1 - -	1 - -	171 - -	207 - -	16
203.	24378	216	209	1 - -	1 - -	192 - -	204 - -	12
204.	24379	216	214	1 - -	1 - -	204 - -	254 - -	9
205.	24380	217	190	14 - -	1 - -	186 - -	231 - -	21
206.	24381	218	195	1 - -	1 - -	206 - -	213 - -	10
207.	24382	215	218	1 - -	1 - -	206 - -	211 - -	6
208.	24383	184	219	1 - -	1 - -	224 - -	170 - -	16

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Lt Face Id T1 id	Rt Edge Id T1 id	Lt Edge Id T1 id	No. Coord
209.	24384	198	220	1 - -	1 - -	211 - -	209 - - 7
210.	24385	221	221	1 - -	15 - -	210 - -	210 - - 9
211.	24386	215	220	1 - -	1 - -	219 - -	202 - - 10
212.	24387	205	223	1 - -	1 - -	212 - -	212 - - 9
213.	24388	218	224	1 - -	1 - -	215 - -	207 - - 4
214.	24389	225	222	1 - -	1 - -	214 - -	255 - - 2
215.	24390	226	224	1 - -	1 - -	248 - -	217 - - 2
216.	24391	228	185	1 - -	1 - -	175 - -	229 - - 12
217.	24392	226	228	1 - -	1 - -	216 - -	258 - - 8
218.	24393	230	211	1 - -	1 - -	196 - -	236 - - 10
219.	24394	229	220	1 - -	1 - -	209 - -	219 - - 7
220.	24395	231	213	1 - -	1 - -	201 - -	227 - - 13
221.	24396	232	230	1 - -	1 - -	218 - -	246 - - 13
222.	24397	233	234	1 - -	1 - -	222 - -	226 - - 2
223.	24398	235	192	1 - -	1 - -	178 - -	266 - - 19
224.	24399	219	236	1 - -	1 - -	225 - -	208 - - 9
225.	24400	235	236	1 - -	1 - -	226 - -	223 - - 2
226.	24401	236	233	1 - -	1 - -	222 - -	224 - - 13
227.	24402	231	237	1 - -	1 - -	227 - -	220 - - 2
228.	24403	238	232	1 - -	1 - -	221 - -	267 - - 12
229.	24404	228	238	1 - -	1 - -	228 - -	217 - - 6
230.	24405	239	153	1 - -	1 - -	230 - -	230 - - 24
231.	24406	217	241	1 - -	16 - -	232 - -	2 - - 6
232.	24407	241	242	1 - -	17 - -	233 - -	3 - - 2
233.	24408	245	242	18 - -	1 - -	8 - -	234 - - 2
234.	24409	245	246	1 - -	19 - -	238 - -	14 - - 2
235.	24410	247	227	1 - -	1 - -	235 - -	242 - - 11
236.	24411	230	248	1 - -	1 - -	269 - -	221 - - 11
237.	24412	251	202	1 - -	14 - -	185 - -	5 - - 28
238.	24413	246	252	1 - -	1 - -	238 - -	253 - - 3
239.	24414	253	244	1 - -	1 - -	239 - -	270 - - 2
240.	24415	254	251	1 - -	21 - -	237 - -	245 - - 3
241.	24416	247	255	1 - -	1 - -	262 - -	235 - - 6
242.	24417	247	240	1 - -	1 - -	242 - -	241 - - 13
243.	24418	248	256	1 - -	1 - -	243 - -	236 - - 10
244.	24419	258	254	1 - -	1 - -	240 - -	252 - - 3
245.	24420	254	260	21 - -	1 - -	9 - -	244 - - 3
246.	24421	232	262	1 - -	1 - -	246 - -	228 - - 6
247.	24422	260	263	20 - -	1 - -	10 - -	245 - - 3
248.	24423	224	261	1 - -	1 - -	260 - -	213 - - 18
249.	24424	264	258	1 - -	23 - -	244 - -	251 - - 4
250.	24425	265	189	1 - -	1 - -	250 - -	250 - - 29
251.	24426	266	264	1 - -	23 - -	249 - -	252 - - 4
252.	24427	258	266	1 - -	23 - -	261 - -	249 - - 6
253.	24428	246	270	1 - -	19 - -	254 - -	234 - - 6
254.	24429	270	216	1 - -	1 - -	203 - -	259 - - 26
255.	24430	271	225	1 - -	1 - -	214 - -	256 - - 6
256.	24431	271	255	1 - -	1 - -	241 - -	272 - - 16
257.	24432	263	272	24 - -	1 - -	263 - -	247 - - 3

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Li Face Id T1 id	Rt Edge Id T1 id	Li Edge Id T1 id	No. Coord
258. 24433	273	226	1	1	215	258	12
259. 24434	270	276	1	19	264	253	3
260. 24435	261	277	1	1	260	248	3
261. 24436	274	266	1	1	251	261 458 286	6
262. 24437	255	278	1	1	268	256	8
263. 24438	279	272	1	24	265	16	4
264. 24439	276	279	1	25	263	15	6
265. 24440	272	280	1	1	265	257	8
266. 24441	282	235	1	1	225	270	15
267. 24442	238	284	1	1	267	229	8
268. 24443	278	285	1	1	268	262	4
269. 24444	288	248	1	1	243	271	16
270. 24445	282	253	1	1	239	288	21
271. 24446	289	288	1	1	269	271	3
272. 24447	271	290	1	1	286	255	19
273. 24448	290	291	1	1	273 458 289	272	19
274. 24449	294	295	1	1	275	274 458 293	3
275. 24450	293	295	26	1	278	289	6
276. 24451	293	298	1	27	277	17	3
277. 24452	298	297	1	1	280	284	2
278. 24453	295	299	26	1	282	274	4
279. 24454	299	300	1	1	279 458 296	278	2
280. 24455	301	297	28	1	281	298	5
281. 24456	297	301	28	1	280	277	3
282. 24457	296	299	1	26	279	17	4
283. 24458	302	296	1	27	282	284	2
284. 24459	302	298	27	1	276	295	3
285. 24460	303	286	1	1	285	312	16
286. 24461	290	306	1	1	287	273	8
287. 24462	306	303	1	1	285	293	15
288. 24463	282	307	1	1	311	266	10
289. 24464	293	308	1	1	289	276	11
290. 24465	287	309	1	1	291	290	10
291. 24466	305	309	1	1	312	291	3
292. 24467	310	275	1	1	292	292	18
293. 24468	306	311	1	1	315	286	2
294. 24469	304	312	1	1	294	294	6
295. 24470	302	313	1	1	295	283	11
296. 24471	307	243	1	1	296	288	38
297. 24472	314	316	29	1	300	302	2
298. 24473	317	301	1	1	281	298	13
299. 24474	283	318	1	1	316	299	10
300. 24475	314	316	1	29	301	297	9
301. 24476	316	319	1	1	335	297	2
302. 24477	314	321	1	1	309	300	3
303. 24478	318	292	1	1	303	299	19
304. 24479	323	322	30	1	306	305	2
305. 24480	319	323	1	1	306	301	2
306. 24481	322	323	30	1	304	304	4

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T1 id	Lt Face Id T1 id	Rt Edge Id T1 id	Lt Edge Id T1 id	No. Coord	
307.	24482	326	320	1 - -	1 - -	307 - -	310 - -	2
308.	24483	324	327	1 - -	1 - -	309 - -	308 458 298	3
309.	24484	327	321	1 - -	1 - -	302 - -	322 - -	2
310.	24485	326	328	1 - -	1 - -	323 - -	307 - -	2
311.	24486	329	307	1 - -	1 - -	296 - -	313 - -	14
312.	24487	309	303	1 - -	1 - -	287 - -	290 - -	23
313.	24488	329	281	1 - -	1 - -	313 - -	345 - -	27
314.	24489	331	330	1 - -	1 - -	314 - -	324 - -	2
315.	24490	311	332	1 - -	1 - -	318 - -	332 - -	6
316.	24491	318	333	1 - -	1 - -	317 - -	303 - -	8
317.	24492	333	268	1 - -	1 - -	317 - -	319 - -	19
318.	24493	325	332	1 - -	1 - -	327 - -	318 - -	9
319.	24494	334	333	1 - -	1 - -	316 - -	319 - -	2
320.	24495	329	336	1 - -	1 - -	321 - -	311 - -	11
321.	24496	336	338	1 - -	1 - -	321 - -	320 - -	2
322.	24497	327	339	1 - -	1 - -	341 - -	308 - -	6
323.	24498	345	328	1 - -	1 - -	310 - -	324 - -	6
324.	24499	345	331	1 - -	1 - -	314 - -	331 - -	8
325.	24500	315	346	1 - -	1 - -	328 - -	325 - -	20
326.	24501	342	346	1 - -	1 - -	325 - -	326 - -	14
327.	24502	332	340	1 - -	1 - -	327 - -	329 - -	15
328.	24503	347	346	1 - -	1 - -	326 - -	328 - -	3
329.	24504	332	350	1 - -	1 - -	329 - -	315 - -	8
330.	24505	343	351	1 - -	1 - -	330 - -	330 - -	5
331.	24506	353	345	1 - -	1 - -	323 - -	333 - -	5
332.	24507	311	354	1 - -	1 - -	332 - -	293 - -	18
333.	24508	353	357	1 - -	31 - -	347 - -	334 - -	3
334.	24509	357	353	1 - -	31 - -	331 - -	333 - -	7
335.	24510	319	358	1 - -	1 - -	335 - -	305 - -	19
336.	24511	349	361	1 - -	1 - -	338 - -	336 - -	6
337.	24512	361	344	1 - -	1 - -	337 - -	336 - -	19
338.	24513	362	361	1 - -	1 - -	337 - -	338 - -	2
339.	24514	363	339	1 - -	1 - -	322 - -	339 - -	19
340.	24515	365	359	1 - -	1 - -	340 - -	340 - -	6
341.	24516	366	339	1 - -	1 - -	339 - -	341 - -	14
342.	24517	348	367	1 - -	1 - -	343 - -	342 - -	17
343.	24518	367	337	1 - -	1 - -	343 - -	353 - -	24
344.	24519	360	369	1 - -	1 - -	376 - -	344 - -	7
345.	24520	329	371	1 - -	1 - -	350 - -	320 - -	45
346.	24521	373	335	1 - -	1 - -	346 - -	346 - -	28
347.	24522	374	357	1 - -	1 - -	334 - -	352 - -	14
348.	24523	375	341	1 - -	1 - -	348 - -	348 - -	27
349.	24524	372	376	1 - -	1 - -	349 - -	398 - -	5
350.	24525	371	377	1 - -	1 - -	350 - -	345 - -	3
351.	24526	379	374	1 - -	32 - -	347 - -	352 - -	3
352.	24527	379	374	32 - -	1 - -	351 - -	354 - -	3
353.	24528	380	367	1 - -	1 - -	342 - -	354 - -	7
354.	24529	379	380	1 - -	1 - -	363 - -	351 - -	2
355.	24530	382	352	1 - -	1 - -	355 - -	355 - -	17

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T'l id	Lt Face Id T'l id	Rt Edge Id T'l id	Lt Edge Id T'l id	No. Coord	
356.	24531	383	355	1 - -	1 - -	356 - -	361 - -	13
357.	24532	356	384	1 - -	1 - -	360 - -	357 - -	28
358.	24533	370	378	1 - -	1 - -	358 - -	358 458 301	19
359.	24534	364	385	1 - -	1 - -	359 456 225	359 - -	15
360.	24535	384	388	1 - -	1 - -	361 - -	357 - -	3
361.	24536	388	383	1 - -	1 - -	356 - -	362 - -	3
362.	24537	389	388	1 - -	1 - -	360 - -	364 - -	2
363.	24538	390	380	1 - -	1 - -	353 - -	363 - -	2
364.	24539	389	391	1 - -	33 - -	366 - -	365 - -	2
365.	24540	392	389	1 - -	33 - -	362 - -	367 - -	4
366.	24541	391	393	1 - -	33 - -	369 - -	364 - -	2
367.	24542	394	392	1 - -	33 - -	365 - -	18 - -	4
368.	24543	369	396	1 - -	1 - -	368 - -	344 - -	13
369.	24544	393	397	1 - -	34 - -	377 - -	18 - -	3
370.	24545	394	397	34 - -	1 - -	369 - -	367 - -	4
371.	24546	398	368	1 - -	1 - -	371 - -	375 - -	13
372.	24547	395	399	1 - -	1 - -	372 458 304	372 - -	9
373.	24548	386	400	1 - -	1 - -	373 - -	373 - -	5
374.	24549	401	398	1 - -	35 - -	371 - -	375 - -	2
375.	24550	398	401	1 - -	35 - -	378 - -	374 - -	7
376.	24551	369	404	1 - -	1 - -	382 - -	368 - -	41
377.	24552	405	397	1 - -	1 - -	370 - -	378 - -	4
378.	24553	405	401	1 - -	1 - -	374 - -	379 - -	3
379.	24554	406	405	1 - -	1 - -	383 - -	379 - -	2
380.	24555	407	402	1 - -	1 - -	380 - -	385 - -	2
381.	24556	408	404	1 - -	36 - -	376 - -	382 - -	3
382.	24557	408	404	36 - -	1 - -	381 - -	383 - -	3
383.	24558	408	405	1 - -	1 - -	377 - -	381 - -	5
384.	24559	411	381	1 - -	1 - -	384 - -	385 - -	11
385.	24560	411	407	1 - -	1 - -	380 - -	411 - -	6
386.	24561	414	409	1 - -	1 - -	386 - -	387 - -	7
387.	24562	414	387	1 - -	1 - -	387 - -	391 - -	10
388.	24563	415	410	1 - -	1 - -	388 - -	393 - -	4
389.	24564	415	416	1 - -	1 - -	389 - -	388 - -	4
390.	24565	418	403	1 - -	1 - -	390 - -	395 - -	34
391.	24566	414	420	1 - -	1 - -	392 - -	386 - -	8
392.	24567	420	412	1 - -	1 - -	392 - -	399 - -	13
393.	24568	415	421	1 - -	1 - -	393 - -	389 - -	8
394.	24569	413	413	1 - -	37 - -	394 - -	394 - -	23
395.	24570	418	423	1 - -	38 - -	397 - -	396 - -	3
396.	24571	423	418	1 - -	38 - -	390 - -	395 - -	9
397.	24572	423	424	1 - -	1 - -	397 - -	396 - -	3
398.	24573	425	372	1 - -	1 - -	349 - -	398 456 263	44
399.	24574	429	420	1 - -	1 - -	391 - -	399 458 306	10
400.	24575	430	427	1 - -	1 - -	400 - -	400 - -	14
401.	24576	431	419	1 - -	1 - -	401 - -	408 - -	9
402.	24577	433	432	1 - -	1 - -	402 - -	402 458 309	13
403.	24578	435	434	1 - -	40 - -	405 - -	405 - -	9

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T'l id	Lt Face Id T'l id	Rt Edge Id T'l id	Lt Edge Id T'l id	No. Coord
404.	24579	417	436	1 - -	1 - -	404 - -	10
405.	24580	434	435	1 - -	40 - -	403 - -	6
406.	24581	438	438	1 - -	39 - -	406 - -	22
407.	24582	437	439	1 - -	1 - -	417 - -	3
408.	24583	443	431	1 - -	1 - -	401 - -	10
409.	24584	445	442	1 - -	1 - -	409 - -	2
410.	24585	446	446	1 - -	41 - -	410 - -	28
411.	24586	411	447	1 - -	1 - -	411 - -	29
412.	24587	445	448	1 - -	1 - -	420 - -	2
413.	24588	422	449	1 - -	1 - -	413 - -	13
414.	24589	451	441	1 - -	1 - -	414 - -	17
415.	24590	428	452	1 - -	1 - -	421 - -	15
416.	24591	453	426	1 - -	1 - -	416 - -	18
417.	24592	439	453	1 - -	1 - -	424 - -	10
418.	24593	454	452	1 - -	1 - -	415 - -	4
419.	24594	454	454	1 - -	43 - -	418 - -	10
420.	24595	455	448	1 - -	1 - -	412 - -	4
421.	24596	456	452	1 - -	1 - -	418 - -	4
422.	24597	440	458	1 - -	1 - -	437 - -	7
423.	24598	459	459	42 - -	1 - -	423 - -	23
424.	24599	460	453	1 - -	1 - -	416 - -	6
425.	24600	450	461	1 - -	1 - -	425 - -	13
426.	24601	444	462	1 - -	1 - -	430 - -	12
427.	24602	463	464	1 - -	1 - -	427 - -	2
428.	24603	466	458	1 - -	44 - -	422 - -	10
429.	24604	467	463	1 - -	1 - -	427 456 310	7
430.	24605	462	468	1 - -	1 - -	431 - -	13
431.	24606	466	468	1 - -	1 - -	448 - -	2
432.	24607	471	469	1 - -	1 - -	432 - -	10
433.	24608	473	462	1 - -	1 - -	426 - -	14
434.	24609	473	472	1 - -	1 - -	436 - -	4
435.	24610	474	465	1 - -	1 - -	435 - -	17
436.	24611	475	472	1 - -	1 - -	434 - -	7
437.	24612	458	477	1 - -	44 - -	439 - -	12
438.	24613	478	476	1 - -	1 - -	438 - -	2
439.	24614	477	479	1 - -	45 - -	444 - -	2
440.	24615	480	466	1 - -	45 - -	431 - -	14
441.	24616	479	480	60 - -	45 - -	445 - -	2
442.	24617	482	473	1 - -	1 - -	434 - -	5
443.	24618	483	470	1 - -	1 - -	443 - -	5
444.	24619	479	484	1 - -	60 - -	447 - -	4
445.	24620	485	480	1 - -	60 - -	440 - -	2
446.	24621	484	485	61 - -	60 - -	452 - -	2
447.	24622	486	484	61 - -	1 - -	446 - -	2
448.	24623	487	468	1 - -	1 - -	430 - -	13
449.	24624	482	489	1 - -	1 - -	449 - -	6
450.	24625	490	486	61 - -	1 456 79	447 - -	2
451.	24626	491	457	1 - -	1 - -	451 - -	32

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T'l id	Lt Face Id T'l id	Rt Edge Id T'l id	Lt Edge Id T'l id	No. Coord
452.	24627	485	493	61 - -	1 - -	453 - -	445 - - 5
453.	24628	493	490	61 - -	1 - -	450 456 405	459 - - 4
454.	24629	494	488	1 - -	1 - -	454 - -	458 - - 6
455.	24630	496	478	1 - -	1 - -	438 - -	455 - - 7
456.	24631	481	497	1 - -	1 - -	457 - -	456 - - 9
457.	24632	492	497	1 - -	1 - -	460 - -	457 - - 9
458.	24633	499	494	1 - -	1 - -	454 - -	458 - - 11
459.	24634	500	493	1 - -	1 - -	452 - -	459 - - 11
460.	24635	501	497	1 - -	1 - -	456 - -	461 - - 10
461.	24636	501	502	1 - -	1 - -	465 - -	463 - - 2
462.	24637	504	495	1 - -	1 - -	462 - -	463 - - 17
463.	24638	501	504	1 - -	1 - -	466 - -	460 - - 12
464.	24639	498	505	1 - -	1 - -	464 - -	464 456 435 16
465.	24640	502	503	1 - -	1 - -	465 - -	467 - - 18
466.	24641	504	507	1 - -	1 - -	466 - -	462 - - 13
467.	24642	508	502	1 - -	1 - -	461 - -	467 - - 23
468.	24643	510	509	1456 1	1 - -	468 - -	477 - - 2
469.	24644	511	506	1 - -	1 - -	469 - -	469 - - 48
470.	24645	514	512	1 - -	1 - -	470 - -	470 - - 6
471.	24646	516	513	1 - -	1 - -	471 - -	471 - - 4
472.	24647	515	515	1 - -	46 - -	472 - -	472 - - 59
473.	24648	518	519	47 - -	1 - -	474 - -	476 - - 2
474.	24649	519	520	47 - -	1 - -	475 - -	473 - - 2
475.	24650	520	521	47 - -	1 - -	476 - -	474 - - 2
476.	24651	521	518	47 - -	1 - -	473 - -	475 - - 34
477.	24652	524	510	1 - -	1 - -	468 - -	477 - - 22
478.	24653	523	525	1 - -	1 - -	480 - -	478 - - 4
479.	24654	525	526	1 - -	1 - -	481 - -	478 - - 2
480.	24655	525	527	1 - -	1 - -	480 - -	479 - - 3
481.	24656	526	528	1 - -	1 - -	481 - -	483 - - 4
482.	24657	530	529	1 - -	48 - -	485 - -	491 - - 4
483.	24658	533	526	1 - -	1 - -	479 - -	483 - - 10
484.	24659	534	529	48 - -	1 - -	482 - -	489 - - 15
485.	24660	535	529	1 - -	1 - -	484 - -	485 - - 22
486.	24661	537	532	1 - -	1 - -	486 - -	487 - - 6
487.	24662	537	536	1 - -	1 - -	487 - -	488 - - 7
488.	24663	537	538	1 - -	1 - -	488 - -	486 - - 5
489.	24664	539	534	48 - -	1 - -	484 - -	494 - - 4
490.	24665	540	517	1 - -	1 - -	490 456 595	490 - - 23
491.	24666	541	530	1 - -	48 - -	492 - -	493 - - 7
492.	24667	545	530	1 - -	1 - -	482 - -	492 - - 10
493.	24668	546	541	1 - -	48 - -	491 - -	496 - - 4
494.	24669	539	547	1 - -	48 - -	497 - -	489 - - 5
495.	24670	548	531	1 - -	1 - -	495 - -	500 - - 6
496.	24671	549	546	1 - -	48 - -	504 - -	497 - - 2
497.	24672	547	549	1 - -	48 - -	505 - -	494 - - 5
498.	24673	550	522	1 - -	1 - -	498 - -	500 - - 19
499.	24674	542	543	1 - -	1 - -	499 - -	499 - - 14

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T'1 id	Lt Face Id T'1 id	Rt Edge Id T'1 id	Lt Edge Id T'1 id	No. Coord	
500.	24675	550	548	1 - -	1 - -	495 - -	503 - -	11
501.	24676	553	552	49 - -	1 - -	507 - -	502 - -	2
502.	24677	553	554	1 - -	49 - -	508 - -	501 - -	3
503.	24678	550	555	1 - -	1 - -	509 - -	498 - -	8
504.	24679	556	546	1 - -	1 - -	493 - -	504 - -	23
505.	24680	555	549	1 - -	1 - -	496 - -	503 - -	11
506.	24681	558	544	1 - -	1 - -	506 - -	512 - -	9
507.	24682	559	552	1 - -	49 - -	501 - -	511 - -	5
508.	24683	560	554	49 - -	1 - -	502 - -	510 - -	7
509.	24684	561	555	1 - -	1 - -	505 - -	515 - -	12
510.	24685	560	562	1 - -	49 - -	525 - -	508 - -	2
511.	24686	559	562	49 - -	1 - -	20 - -	507 - -	4
512.	24687	563	558	1 - -	1 - -	506 - -	519 - -	5
513.	24688	551	564	1 - -	1 - -	513 - -	513 - -	18
514.	24689	565	563	1 - -	1 - -	512 - -	520 - -	6
515.	24690	567	561	50 - -	1 - -	517 - -	516 - -	11
516.	24691	566	567	1 - -	1 - -	518 - -	516 456 730	2
517.	24692	569	561	1 - -	50 - -	509 - -	521 - -	13
518.	24693	570	567	1 - -	1 - -	522 - -	518 456 739	4
519.	24694	563	569	1 - -	1 - -	517 - -	514 - -	18
520.	24695	565	571	1 - -	51 - -	524 - -	529 - -	8
521.	24696	572	569	1 - -	50 - -	519 - -	522 - -	2
522.	24697	567	572	1 - -	50 - -	536 - -	515 - -	5
523.	24698	574	573	52 - -	1 - -	533 - -	526 - -	4
524.	24699	575	571	51 - -	1 - -	520 - -	544 - -	3
525.	24700	562	576	1 - -	1 - -	525 - -	511 - -	20
526.	24701	574	577	1 - -	52 - -	527 - -	523 - -	5
527.	24702	578	577	52 - -	1 - -	526 - -	531 - -	3
528.	24703	579	575	51 - -	1 - -	524 - -	539 - -	3
529.	24704	580	565	1 - -	51 - -	514 - -	530 - -	10
530.	24705	581	580	1 - -	51 - -	529 - -	543 - -	3
531.	24706	578	582	1 - -	52 - -	532 - -	527 - -	6
532.	24707	582	583	1 - -	52 - -	541 - -	531 - -	2
533.	24708	573	583	52 - -	1 - -	532 - -	523 - -	23
534.	24709	568	584	1 - -	1 - -	534 - -	534 - -	17
535.	24710	585	586	1 - -	62 - -	538 456 760	537 - -	2
536.	24711	585	572	1 - -	1 - -	521 - -	535 - -	16
537.	24712	587	585	1 - -	62 - -	536 - -	538 456 762	2
538.	24713	587	586	62 - -	1 456 80	535 - -	537 - -	2
539.	24714	590	579	51 - -	1 - -	528 - -	547 - -	11
540.	24715	593	595	1 - -	1 - -	540 456 783	544 - -	2
541.	24716	589	583	1 - -	1 - -	533 - -	541 - -	20
542.	24717	596	597	1 - -	51 - -	543 - -	545 - -	2
543.	24718	581	597	51 - -	1 - -	542 - -	530 - -	10
544.	24719	575	593	1 - -	1 - -	540 - -	528 - -	41
545.	24720	591	596	1 - -	51 - -	542 - -	549 - -	10
546.	24721	592	598	51 - -	1 - -	547 - -	551 - -	7
547.	24722	598	590	51 - -	1 - -	539 - -	546 - -	10

(table continued)

APPENDIX H

TABLE 93. Edge Records From Tile NJ32 (Con't)

ID LINE ID	SN	EN	Rt Face Id T'l id	Lt Face Id T'l id	Rt Edge Id T'l id	Lt Edge Id T'l id	No. Coord
548.	24723	588	594	1 - -	1 - -	548 - -	9
549.	24724	602	591	1 - -	51 - -	545 - -	20
550.	24725	603	600	1 - -	1 - -	550 456 794	4
551.	24726	602	592	51 - -	1 - -	546 - -	21
552.	24727	606	605	1 - -	53 - -	558 - -	2
553.	24728	607	601	1 - -	1 - -	553 - -	9
554.	24729	599	604	1 - -	1 - -	554 - -	32
555.	24730	606	610	53 - -	1 - -	556 - -	9
556.	24731	613	610	1 - -	53 - -	555 - -	5
557.	24732	613	614	53 - -	1 - -	559 - -	2
558.	24733	605	615	1 - -	53 - -	559 - -	26
559.	24734	615	614	1 - -	53 - -	557 - -	20
560.	24735	608	617	1 - -	1 - -	560 - -	10
561.	24736	616	619	1 - -	1 - -	561 - -	6
562.	24737	621	621	1 - -	57 - -	562 - -	9
563.	24738	623	618	1 - -	1 - -	563 - -	7
564.	24739	620	625	1 - -	1 - -	567 - -	5
565.	24740	626	627	1 - -	59 - -	566 - -	3
566.	24741	622	627	1 - -	1 - -	568 - -	7
567.	24742	626	625	59 - -	1 - -	569 - -	5
568.	24743	629	627	59 - -	1 - -	565 - -	12
569.	24744	625	629	59 - -	1 - -	568 - -	14
570.	24745	630	624	1 - -	1 - -	570 - -	4
571.	24746	628	632	1 - -	58 - -	573 - -	13
572.	24747	628	633	58 - -	1 - -	573 - -	19
573.	24748	632	633	1 - -	58 - -	572 - -	8
574.	24749	635	631	1 - -	1 - -	574 - -	13
575.	24750	638	634	1 - -	1 - -	575 - -	10
576.	24751	639	636	1 - -	1 - -	576 - -	8
577.	24752	637	640	1 - -	1 - -	577 456 908	9

APPENDIX H

TABLE 94. Face Records for Tile NJ32

The header file...

```
;Face Primitives;-;ID=1,          1,P,Row Identifier,-,-,
;DNAREA.AFT_ID=I,                1,F,Foreign Key to Area Feature table,-,-,
RING_PTR=I,                       1,F,Foreign Key to Ring table,-,-,
```

The actual data...

1.	4570	1	2.	4571	37	3.	4572	38	4.	4573	39
5.	4574	40	6.	4575	41	7.	4576	42	8.	4577	43
9.	4578	44	10.	4579	45	11.	4580	46	12.	4581	47
13.	4582	48	14.	4583	49	15.	4584	50	16.	4585	51
17.	4586	52	18.	4587	53	19.	4588	54	20.	4589	55
21.	4590	56	22.	4591	57	23.	4592	58	24.	4593	59
25.	4594	60	26.	4595	61	27.	4596	62	28.	4597	63
29.	4598	64	30.	4599	65	31.	4600	66	32.	4601	67
33.	4602	68	34.	4603	69	35.	4604	70	36.	4605	71
37.	4606	72	38.	4607	73	39.	4608	74	40.	4609	75
41.	4610	76	42.	4611	77	43.	4612	78	44.	4613	79
45.	4614	80	46.	4615	81	47.	4616	82	48.	4617	83
49.	4618	84	50.	4619	85	51.	4620	86	52.	4621	87
53.	4622	88	54.	4623	92	55.	4624	93	56.	4625	94
57.	4626	95	58.	4627	96	59.	4628	97	60.	4629	98
61.	4630	99	62.	4631	100						

TABLE 95. Ring Records for NJ32

The header file...

```
;Ring table;-;ID=I,          1, P, Row ID,-,-,
;FACE_ID=I                    1, F, Foreign key to face table,-,-,
;START_EDGE=I,                1, F, Foreign Key to Edge table,-,-,
;:
```

The actual data...

1.	1	36	2.	1	75	3.	1	77	4.	1	132
5.	1	137	6.	1	151	7.	1	162	8.	1	173
9.	1	186	10.	1	210	11.	1	275	12.	1	297
13.	1	304	14.	1	333	15.	1	364	16.	1	374
17.	1	381	18.	1	394	19.	1	395	20.	1	403
21.	1	406	22.	1	410	23.	1	419	24.	1	423
25.	1	428	26.	1	472	27.	1	473	28.	1	482
29.	1	501	30.	1	515	31.	1	520	32.	1	523
33.	1	552	34.	1	562	35.	1	565	36.	1	571
37.	2	36	38.	3	75	39.	4	77	40.	5	89
41.	6	1	42.	7	96	43.	8	1	44.	9	137
45.	10	132	46.	11	151	47.	12	162	48.	13	173

(table continued)

APPENDIX H

TABLE 95. Ring Records for NJ32 (Cont)

49.	14	2	50.	15	210	51.	16	2	52.	17	3
53.	18	8	54.	19	14	55.	20	5	56.	21	9
57.	22	7	58.	23	249	59.	24	11	60.	25	13
61.	26	17	62.	27	17	63.	28	280	64.	29	297
65.	30	304	66.	31	333	67.	32	351	68.	33	18
69.	34	18	70.	35	374	71.	36	381	72.	37	394
73.	38	395	74.	39	406	75.	40	403	76.	41	410
77.	42	423	78.	43	419	79.	44	19	80.	45	19
81.	46	472	82.	47	473	83.	48	482	84.	49	501
85.	50	515	86.	51	520	87.	52	523	88.	53	552
89.	53	21	90.	53	22	91.	53	23	92.	54	21
93.	55	23	94.	56	22	95.	57	562	96.	58	571
97.	59	565	98.	60	441	99.	61	446	100.	62	535

TABLE 96. Entity Node records for NJ32

The header file...

```

;Entity Node Primitives;-;ID=I,      1,P,Row Identifier,--,
:DNPOINT.PFT_ID=I                    1,F,Foreign Key to Point Feature table,-,-,
,
:CONTAINING_FACE=I,                  1,F,Foreign Key to Face table,-,-,
:FIRST_EDGE=X,                        1,N,Foreign Key to Edge table (null),-,-,
:COORDINATE=C,                        1,N,Coordinates of Node,-,-,

```

The actual data...

1	345	1	13.563	37.843
2	346	9	12.752	37.694
3	347	1	14.609	37.673
4	348	1	13.550	37.650
5	349	1	10.023	36.541
6	350	61	10.009	36.524

TABLE 97. Text records for NJ32

The header file.

```

Text Primitives;-;ID=I, 1,P,Row Identifier,-,-,
:DNTEXT.TFT_ID=I,      1,F,Foreign Key to Text Feature table,-,-,
:STRING=T,             *,N,Text String,-,-,
:SHAPE_LINE=C,        *,N,Shape of Text String,-,-,
;;

```

(table continued)

APPENDIX H

TABLE 97. Text records for NJ32 (Cont)

The actual data...

1	529Fiume (stream) Salso			
	14.405	37.736	14.448	37.729
	14.522	37.687	14.603	37.686
2	530 Simeto			
	14.803	37.715	14.807	37.684
	14.805	37.665	14.808	37.650
3	531 Belice			
	12.908	37.686	12.934	37.712
	12.954	37.730		
4	532 Dittaino			
	14.509	37.560	14.544	37.550
	14.580	37.556	14.616	37.570
5	533 Fiume Salso			
	13.913	37.149	13.921	37.169
	13.968	37.195	13.988	37.223
6	534 underground			
	10.472	35.729	10.498	35.747
	10.533	35.772	10.559	35.790
7	535 aqueduct			
	10.495	35.725	10.558	35.767
8	536 Sebkra de Sidiel	Hani		
	10.313	35.560	10.520	35.565
9	537 (marsh)			
	10.314	35.540	10.379	35.539

TABLE 98. Feature Spatial Index for Tile NJ32

The number of features = 62

The MBR is 10.00 35.00 - 15.00 38.19 the number of nodes = 63

The Director follows...

Node	Offset	No. Entries	Total
1	0	10	10
2	80	1	11
3	88	3	14
4	112	8	22
6	176	5	27
7	216	1	28
8	224	3	31
9	248	2	33
15	264	1	34
16	272	2	36
17	288	8	44
30	352	4	48

(table continued)

APPENDIX H

TABLE 98. Feature Spatial Index for Tile NJ32 (Cont)

31	384	2	50
34	400	2	52
35	416	2	54
62	432	3	57
63	456	5	62

The indexed features in one byte MBR locations...

Row Id	XMIN	YMIN	XMAX	YMAX	Feature Id
1.	173	218	174	220	10
2.	0	0	255	255	1
3.	167	237	170	240	2
4.	231	226	234	229	3
5.	178	222	182	228	4
6.	236	224	238	225	5
7.	238	223	239	225	6
8.	237	224	239	225	7
9.	238	223	240	225	8
10.	139	215	141	220	9
11.	250	194	252	197	20
12.	9	64	18	75	48
13.	1	123	8	138	44
14.	1	123	8	132	45
15.	250	194	252	197	21
16.	246	193	249	197	19
17.	247	194	250	197	18
18.	248	194	251	198	17
19.	248	196	250	202	16
20.	219	200	221	202	15
21.	245	196	251	210	14
22.	231	212	236	215	11
23.	10	151	15	157	37
24.	9	144	16	148	39
25.	47	145	50	147	40
26.	9	141	15	145	41

The indexed features in one byte MBR locations...

Row Id	XMIN	YMIN	XMAX	YMAX	Feature Id
27.	6	137	11	144	42
28.	15	27	17	29	56
29.	218	175	220	178	31
30.	249	192	252	195	24
31.	247	192	250	195	25
32.	155	210	158	212	13
33.	180	212	182	214	12
34.	3	18	5	20	57

(table continued)

APPENDIX H

TABLE 98. Feature Spatial Index for Tile NJ32 (Cont)

35.	249	194	251	196	22
36.	252	194	254	195	23
37.	222	159	224	161	36
38.	253	184	254	185	30
39.	251	185	254	187	29
40.	249	189	251	190	28
41.	250	189	253	191	27
42.	251	189	255	191	26
43.	225	162	227	165	33
44.	224	162	226	164	34
45.	0	118	2	123	61
46.	0	121	2	124	60
47.	21	82	26	102	46
48.	21	78	28	85	47
49.	0	42	1	43	62
50.	15	32	32	52	51
51.	251	138	253	140	43
52.	227	150	230	153	38
53.	218	167	219	169	32
54.	220	161	222	163	35
55.	38	56	41	61	49
56.	46	43	51	51	52
57.	36	11	43	15	59
58.	2	49	10	58	50
59.	12	23	20	32	53
60.	16	28	18	30	54
61.	14	26	16	30	55
62.	22	8	31	16	58

APPENDIX H

TABLE 99. Face MBR Records for NJ32

The header file...

```

; "FBR
  Face Bounding Rectangle table";-;ID=I, 1, P, Row ID,-,-,
:XMIN=F, 1, N, Minimum X Coordinate,-,-,
:YMIN=F, 1, N, Minimum Y Coordinate,-,-,
:XMAX=F, 1, N, Maximum X Coordinate,-,-,
:YMAX=F, 1, N, Maximum Y Coordinate,-,-,
;

```

The actual data...

1.	10.00	35.00	15.00	38.19	2.	13.28	37.97	13.31	37.99
3.	14.54	37.83	14.58	37.85	4.	13.51	37.78	13.56	37.85
5.	14.64	37.80	14.65	37.81	6.	14.67	37.80	14.68	37.80
7.	14.66	37.80	14.67	37.80	8.	14.67	37.80	14.69	37.80
9.	12.74	37.69	12.76	37.75	10.	13.40	37.72	13.41	37.74
11.	14.55	37.65	14.61	37.68	12.	13.55	37.65	13.56	37.67
13.	13.05	37.63	13.09	37.65	14.	14.82	37.45	14.92	37.62
15.	14.30	37.51	14.32	37.52	16.	14.87	37.45	14.90	37.52
17.	14.87	37.43	14.91	37.46	18.	14.85	37.43	14.89	37.46
19.	14.82	37.42	14.88	37.46	20.	14.91	37.43	14.92	37.45
21.	14.92	37.43	14.93	37.45	22.	14.88	37.43	14.91	37.44
23.	14.94	37.43	14.97	37.44	24.	14.89	37.41	14.92	37.43
25.	14.85	37.41	14.89	37.43	26.	14.93	37.36	15.00	37.38
27.	14.92	37.36	14.94	37.38	28.	14.89	37.36	14.91	37.37
29.	14.93	37.31	14.98	37.34	30.	14.96	37.30	14.98	37.31
31.	14.29	37.20	14.30	37.22	32.	14.28	37.09	14.29	37.10
33.	14.42	37.04	14.45	37.06	34.	14.40	37.03	14.42	37.04
35.	14.33	37.02	14.35	37.03	36.	14.37	37.00	14.38	37.01
37.	10.20	36.89	10.29	36.96	38.	14.47	36.88	14.50	36.91
39.	10.19	36.81	10.30	36.85	40.	10.93	36.81	10.97	36.83
41.	10.19	36.77	10.28	36.81	42.	10.12	36.72	10.20	36.80
43.	14.93	36.73	14.95	36.75	44.	10.02	36.55	10.16	36.72
45.	10.02	36.54	10.14	36.65	46.	10.42	36.04	10.49	36.27
47.	10.43	35.98	10.53	36.06	48.	10.19	35.81	10.35	35.93
49.	10.75	35.71	10.80	35.75	50.	10.06	35.62	10.18	35.72
51.	10.31	35.41	10.62	35.65	52.	10.91	35.54	11.00	35.63
53.	10.25	35.29	10.39	35.39	54.	10.32	35.35	10.34	35.37
55.	10.28	35.33	10.30	35.36	56.	10.31	35.34	10.32	35.35
57.	10.07	35.23	10.10	35.24	58.	10.45	35.10	10.60	35.19
59.	10.72	35.15	10.83	35.18	60.	10.01	36.52	10.02	36.54
61.	10.00	36.49	10.02	36.52	62.	10.00	35.54	10.00	35.54

INDEX

	<u>PARAGRAPH</u>	<u>PAGE</u>
Area feature	5.3.3.1	57
Area feature class	5.2.2.2.3	30
Attribute accuracy	5.2.3.1	44
Attribute completeness	5.2.3.1	44
Attribute table	5.2.1.2	21
	5.2.1.6	24
Byte order	5.4.1.1	71
Cartographic primitive	5.2.2	25
	5.2.2.1	25
	5.3.2	48
Character value code	5.2.2.3.2	38
Column data type	5.3.1.1	44
Column description	5.3.1.1	44
Column name	5.3.1.1	44
Column, product-specific	5.2.1.3	23
Column, VPF standard specified	5.2.1.3	23
Complex feature class	5.2.2.2.3	30
	5.3.3	57
Connected node primitive	5.2.2.1	26
	5.3.2.1	49
	5.3.3.1	57
Constructing feature class	5.2.2.2.4	30
Coordinate string	5.3.2.2	51
Coordinate system	5.2.2.4	40
Coordinate, syntax	5.5.5	89
Coverage	5.2.2.3	33
	5.3.4	59
Coverage attribute table	5.3.4.1	59
	5.3.5	62
Coverage, tiled	5.2.2.3.3	38
Cross-tile key	5.2.2.3.4	39
Data quality	5.2.3	43
Data quality coverage	5.3.5	62
Data quality reference coverage	5.2.2.4.5	42
Data quality table	5.2.2.4.5	42
	5.2.2.5	42
	5.3.5	62
Data quality, attribute accuracy	5.2.3.1	44
Data quality, attribute completeness	5.2.3.1	44
Data quality, date status	5.2.3.1	44
Data quality, feature completeness	5.2.3.1	44
Data quality, logical consistency	5.2.3.1	44
Data quality, positional accuracy	5.2.3.1	44
Data quality, source	5.2.3.1	44
Database	5.2.2	25
	5.2.2.5	42
Database header table	5.2.2.5	42
Database name	5.3.6.2	67
Date and time, syntax	5.5.3	83
Date status	5.2.3.1	44
Datum code	5.3.5.2	62

INDEX

	<u>PARAGRAPH</u>	<u>PAGE</u>
Datum name	5.3.5.2	62
Directory	5.2	20
Directory name	5.3.1.2	46
Edge coordinate string	5.3.2.2	51
Edge direction	5.2.2.1.2	28
	5.2.2.2	29
Edge primitive	5.2.2.1	26
	5.3.2.2	51
	5.3.3.1	57
End node	5.2.2.1.2	28
Entity node primitive	5.2.2.1	26
	5.3.2.1	49
	5.3.3.1	57
Face primitive	5.2.2.1	26
	5.3.2.3	53
	5.3.3.1	57
Feature	5.2.2	25
Feature class	5.2.2.2.1	29
	5.3.3	57
Feature class schema table	5.2.2.2.4	30
	5.3.4.2	59
Feature class, complex	5.2.2.2.3	30
	5.3.3	57
Feature class, simple	5.2.2.2.3	30
	5.3.3	57
Feature class, text	5.2.2.2.3	30
	5.3.3	57
Feature completeness	5.2.3.1	44
Feature definition	5.2.2.2.1	29
Feature join	5.2.2.2.2	29
	5.3.3.2	58
Feature table	5.2.1.2	21
File	5.2.1.1	20
Geographic reference file	5.2.2.4.3	41
Geographic reference table	5.3.5	62
Geometric primitive	5.2.2	25
	5.2.2.1	26
	5.3.2	48
Index	5.2	20
Index, spatial	5.2.1.4	24
	5.4.2	74
Index, thematic	5.2.1.4	24
	5.4.3	76
Index, variable-length	5.2.1.4	24
	5.4.1.3	73
Inner ring	5.2.2.1.3	28
	5.3.2.3	53
Integer number, syntax	5.5.1	81
Integer value code	5.2.2.3.2	38
Isolated feature	5.2.2.1.1	26
Key type	5.3.1.1	44

INDEX

	<u>PARAGRAPH</u>	<u>PAGE</u>
Left edge	5.2.2.1.2	28
	5.3.2.2	51
Left face	5.2.2.1.2	28
	5.3.2.2	51
Level of topology	5.2.2.3.1	33
Library	5.2.2	25
	5.2.2.4	40
	5.3.5	62
Library attribute table	5.2.2.5	42
Library header file	5.2.2.4.2	41
Library header table	5.3.5	62
Library reference coverage	5.2.2.4.4	42
	5.3.5	62
Line feature	5.3.3.1	57
Line feature class	5.2.2.2.3	30
Linear feature	5.2.2.1.2	28
Logical consistency	5.2.3.1	44
Minimum bounding rectangle	5.3.2.5	55
Names Placement coverage	5.2.2.4.6	42
Naming convention	5.4.5	78
Narrative file	5.2.1.3	23
Narrative table	5.2.1.5	24
Node primitive, connected	5.2.2.1	26
	5.3.2.1	49
Node primitive, entity	5.2.2.1	26
	5.3.2.1	49
One-dimensional primitive	5.2.2.1.2	28
Outer ring	5.2.2.1.3	28
	5.3.2.3	53
Pathname	5.2.1.1	20
Point feature	5.3.3.1	57
Point feature class	5.2.2.2.3	30
Positional accuracy	5.2.3.1	44
Primitive table	5.2.1.2	21
Primitive, cartographic	5.2.2	25
	5.2.2.1	26
	5.3.2	48
Primitive, geometric	5.2.2	25
	5.2.2.1	26
	5.3.2	48
Product-specific column	5.2.1.3	23
Product-specific directory	5.2.2.4	40
	5.2.2.5	42
Product specification	4.3	17
	5.2.1.3	23
	5.2.2.3.3	38
	5.2.2.4	40
	5.2.2.5	42
	5.3.3	57
Projection code	5.3.5.2	62
Projection name	5.3.5.2	62

INDEX

	<u>PARAGRAPH</u>	<u>PAGE</u>
Real number, syntax	5.5.2	82
Right edge	5.2.2.1.2	28
	5.3.2.2	51
Right face	5.2.2.1.2	28
	5.3.2.2	51
Ring	5.2.2.1.3	28
	5.3.2.3	53
Ring table	5.3.2.3	53
Ring, inner	5.2.2.1.3	28
	5.3.2.3	53
Ring, outer	5.2.2.1.3	28
	5.3.2.3	53
Row identifier	5.2.1.3	23
Shape line	5.3.2.4	55
Simple feature class	5.2.2.2.3	30
	5.3.3	57
Source	5.2.3.1	44
Spatial extent	5.2.2.4	40
	5.3.5	62
Spatial index	5.2.1.4	24
	5.4.2	74
Start node	5.2.2.1.2	28
Table	5.2	20
Table definition	5.3.1.1	44
Table name	5.3.1.2	46
Table name suffix	5.3.1.2	46
Text feature	5.3.3.1	57
Text feature class	5.2.2.2.3	30
Text primitive	5.2.2.1	26
	5.3.2.4	55
	5.3.3.1	57
Text string	5.3.2.4	55
Text, syntax	5.5.4	84
Thematic index	5.2.1.4	24
	5.4.3	76
Tile	5.2.2.3.3	38
Tile reference coverage	5.2.2.4.1	41
	5.3.5	62
Tiled coverage	5.2.2.3.3	38
	5.3.3.3	59
Tiling	5.2.2.3.3	38
	5.3.5	62
Tiling scheme	5.2.2.3.3	38
Topology	5.2.2.3.1	33
Triplet id	5.2.2.3.4	39
	5.3.2.2	51
	5.3.3.1	57
	5.3.3.3	59
	5.4.6	80
Two-dimensional primitive	5.2.2.1.3	28
Universe face	5.3.2.3	53

INDEX

	<u>PARAGRAPH</u>	<u>PAGE</u>
Value description table	5.2.1.3	23
	5.2.2.3.2	38
	5.3.4.3	61
Variable-length index	5.2.1.4	24
	5.4.1.3	73
Version control	1.4	1
VPF basic table structure	5.2.1.2	21
VPF compliant	4.1	16
VPF data model	5.2	20
VPF directory	5.2.1.1	20
VPF standard-specified column	5.2.1.3	23
VPF table	5.2.1.2	21
	5.4.1	70
VPF table content	5.2.1.3	23
	5.4.1.2	73
VPF table header	5.2.1.3	23
	5.4.1.1	71
VPF topology	5.2.2.3.1	33
VPF version number	5.3.6.2	67
Winged-edge topology	5.2.2.1.2	28
	5.3.2.2	51
Zero-dimensional primitive	5.2.2.1.1	26

CONCLUDING MATERIAL

Custodians:

DMA - MP

Preparing activity:

DMA - MP

Review activities:

Army - PO

Navy - NO

Air Force - 09

DLA -

Agent:

Environmental Systems

Research Institute, Inc.

(Project MCGT-0041)

User activities:

Army - PO

Navy - NO

Air Force - 09

