

Final Committee Draft ISO/IEC FCD 13249-5	
Date: 1999-08- 17	Reference number: ISO/JTC 1/SC 32 N 0342
Supersedes document SC 32 N 0041	

THIS DOCUMENT IS STILL UNDER STUDY AND SUBJECT TO CHANGE. IT SHOULD NOT BE USED FOR REFERENCE PURPOSES.

ISO/IEC JTC 1/SC 32 Data Management and Interchange Secretariat: USA (ANSI)	<p>Circulated to P- and O-members, and to technical committees and organizations in liaison for voting (P-members only) by:</p> <p style="text-align: center;">1999-12-31</p> <p>Please return all votes and comments in electronic form directly to the SC 32 Secretariat by the due date indicated.</p>
--	--

ISO/IEC FCD 13249-5
 Title: ISO/IEC FCD 13249-5 Information Technology — Database Languages — SQL Multimedia and Application Packages — Part 5: Still Image
 Project: 1.64.05.00.00

Introductory note: The attached document is hereby submitted for a four-month letter ballot to the National Bodies of ISO/IEC JTC 1/SC 32. The ballot starts 1999-08-31; ITTF must have the document two weeks before the balloting can start.

Medium: E
 No. of pages: 57

Address Reply to: Douglas Mann, Secretariat, ISO/IEC JTC 1/SC 32, Pacific Northwest National Laboratory, 901 D Street, SW., Suite 900, Washington, DC, 20024-2115, United States of America
 Telephone: +1 703 575 2114; Facsimile: +1 703 681 9180; E-mail: MannD@battelle.org

SQL/MM SAF-005

Date: 1999-07-15

ISO/IEC FCD 13249-5:199x (E)

ISO/IEC JTC 1/SC 32/WG 4

Secretariat: U.S.A.

**Information Technology — Database Languages —
SQL Multimedia and Application Packages —
Part 5: Still Image**

Contents	Page
FOREWORD	iii
INTRODUCTION	v
1 SCOPE	1
2 NORMATIVE REFERENCES	1
2.1 <i>International standards</i>	<i>1</i>
3 DEFINITIONS, NOTATIONS, AND CONVENTIONS	2
3.1 DEFINITIONS	2
3.1.1 <i>Definitions provided in Part 1</i>	<i>2</i>
3.1.2 <i>Definitions provided in Part 5</i>	<i>2</i>
3.1.3 <i>Definitions taken from ISO/IEC 9075</i>	<i>2</i>
3.2 NOTATIONS	2
3.2.1 <i>Notations provided in Part 1</i>	<i>2</i>
3.2.2 <i>Notations provided in Part 5</i>	<i>2</i>
3.3 CONVENTIONS	2
4 CONCEPTS	3
5 STILL IMAGE TYPES	5
5.1 SI_STILLIMAGE TYPES AND ROUTINES.....	5
5.1.1 <i>SI_StillImage Type</i>	<i>5</i>
5.1.2 <i>SI_StillImage Methods</i>	<i>8</i>
5.1.3 <i>SI_setContent Method</i>	<i>10</i>
5.1.4 <i>SI_commentText Method</i>	<i>11</i>
5.1.5 <i>Private Functions</i>	<i>12</i>
6 FEATURE TYPES	15
6.1 SI_FEATURE TYPE AND ROUTINES	15
6.1.1 <i>SI_Feature Type</i>	<i>15</i>
6.1.2 <i>SI_Score Method</i>	<i>16</i>
6.2 SI_AVERAGECOLOR TYPE AND ROUTINES	17
6.2.1 <i>SI_AverageColor Type</i>	<i>17</i>
6.2.2 <i>SI_AverageColor Method</i>	<i>18</i>
6.2.3 <i>SI_Score Method</i>	<i>19</i>
6.2.4 <i>SI_findAverageColor Function</i>	<i>20</i>
6.3 SI_COLORHISTOGRAM TYPE AND ROUTINES	21
6.3.1 <i>SI_ColorHistogram Type</i>	<i>21</i>
6.3.2 <i>SI_ColorHistogram Method</i>	<i>23</i>
6.3.3 <i>SI_Score Method</i>	<i>25</i>
6.4 SI_POSITIONALCOLOR TYPE AND ROUTINES.....	27
6.4.1 <i>SI_PositionalColor Type</i>	<i>27</i>
6.4.2 <i>SI_Score Method</i>	<i>28</i>
6.4.3 <i>SI_findPositionalColor Function</i>	<i>29</i>
6.5 SI_TEXTURE TYPE AND ROUTINES.....	30
6.5.1 <i>SI_Texture Type</i>	<i>30</i>
6.5.2 <i>SI_Score Method</i>	<i>31</i>
6.5.3 <i>SI_findTexture Function</i>	<i>32</i>

6.6 SI_FEATURELIST TYPE AND ROUTINES	33
6.6.1 <i>SI_FeatureList Type</i>	33
6.6.2 <i>SI_Score Method</i>	35
6.6.3 <i>SI_Append Method</i>	36
6.6.4 <i>SI_FeatureList Method</i>	37
6.7 AUXILIARY TYPES AND ROUTINES.....	38
6.7.1 <i>SI_ColorType Type</i>	38
6.7.2 <i>SI_ColorType Method</i>	39
7 STATUS CODES	41
8 CONFORMANCE.....	43
8.1 CLAIMS OF CONFORMANCE.....	43
ANNEX A	45
ANNEX B	46
INDEX	47

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 13249 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*.

This document is based on the content of ISO/IEC Final Draft International Standard Database Language (SQL).

This is the first edition of ISO/IEC SQL/MM – Part 5: Still Image.

Introduction

The purpose of this International Standard is to define multimedia and application specific types and their associated routines using the user-defined features in ISO/IEC 9075.

SQL/MM is structured as a multi-part standard. At present it consists of the following parts:

Part 1: Framework

Part 2: Full-Text

Part 3: Spatial

Part 5: Still Image

The organization of this part of ISO/IEC 13249 is as follows:

- 1) Clause 1, "Scope", specifies the scope of this part of ISO/IEC 13249.
- 2) Clause 2, "Normative references", identifies additional standards that, through reference in this part of ISO/IEC 13249, constitute provisions of this part of ISO/IEC 13249.
- 3) Clause 3, "Definitions, notations, and conventions", defines the notations and conventions used in this part of ISO/IEC 13249.
- 4) Clause 4, "Concepts", presents concepts used in the definition of this part of ISO/IEC 13249.
- 5) Clause 5, "Still Image Types", defines the still image user-defined types and associated routines.
- 6) Clause 6, "Feature Types", defines the user-defined types provided for the manipulation of still image features.
- 7) Clause 7, "Status Codes", defines the SQLSTATE codes used in this part of ISO/IEC 13249.
- 8) Clause 8, "Conformance", defines the criteria for conformance to this part of ISO/IEC 13249.
- 9) Annex A, "Implementation-defined elements", is an informative Annex. It lists those features for which the body of this part of ISO/IEC 13249 states that the syntax or meaning or effect on the database is partly or wholly implementation-defined, and describes the defining information that an implementor shall provide in each case.
- 10) Annex B, "Implementation-dependent elements", is an informative Annex. It lists those features for which the body of this part of ISO/IEC 13249 states explicitly that the meaning or effect on the database is implementation-dependent.

In the text of this part of ISO/IEC 13249, Clauses begin a new odd-numbered page, and in Clause 5, "Still Image Types", through Clause 8, "Conformance", Subclauses begin a new page. Any resulting blank space is not significant.

Information Technology — Database Languages — SQL Multimedia and Application Packages — SQL/MM Part 5: Still Image

1 Scope

This part of ISO 13249:

- a) introduces the Still Image part of this International Standard,
- b) gives the references necessary for this part of this International Standard,
- c) defines notations and conventions specific to this part of this International Standard,
- d) defines concepts specific to this part of this International Standard,
- e) defines the still image user-defined types and their associated routines.

2 Normative references

The following standards and publicly available specifications contain provisions that, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards and publicly available specifications are subject to revision and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of standards and public specifications listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

2.1 International standards

ISO/IEC 9075-1:1999, *Information Technology — Database Languages — SQL — Part 1: Framework (SQL/Framework)*.

ISO/IEC 9075-2:1999, *Information Technology — Database Languages — SQL — Part 2: Foundation (SQL/Foundation)*.

ISO/IEC 9075-4:1999, *Information Technology — Database Languages — SQL — Part 4: Persistent Stored Modules (SQL/PSM)*.

ISO/IEC 13249-1:1999, *Information Technology — Database Languages — SQL Multimedia and Application Packages — Part 1: Framework*.

3 Definitions, notations, and conventions

3.1 Definitions

For the purpose of this International Standard, the following definitions apply.

3.1.1 Definitions provided in Part 1

This part of ISO/IEC 13249 makes use of all terms defined in Part 1 of ISO/IEC 13249.

3.1.2 Definitions provided in Part 5

This part of ISO/IEC 13249 defines the following terms:

****Editor's Note 5-010****

The required definitions should be provided or this subclause should be deleted.

3.1.3 Definitions taken from ISO/IEC 9075

This part of ISO/IEC 13249 makes use of the following terms defined in ISO/IEC 9075:

- a) mutator method

3.2 Notations

3.2.1 Notations provided in Part 1

The notations used in this part of ISO/IEC 13249 are defined in Part 1 of ISO/IEC 13249.

3.2.2 Notations provided in Part 5

This part of ISO/IEC 13249 uses the prefix 'SI_' for user-defined type, attribute and SQL-invoked routine names.

3.3 Conventions

The conventions used in this part of ISO/IEC 13249 are defined in Part 1 of ISO/IEC 13249.

4 Concepts

To be supplied.

5 Still Image Types

The types in this family provide for the storage and retrieval of still image values.

5.1 SI_StillImage Types and Routines

5.1.1 SI_StillImage Type

Purpose

The *SI_StillImage* type provides the definition of a still image type.

Definition

```

CREATE TYPE SI_StillImage
  AS (
    SI_bitsPerColor INTEGER,
    SI_bitsPerPixel INTEGER,
    SI_comment CHARACTER VARYING(SI_MaxCommentLength),
    SI_content BINARY LARGE OBJECT(SI_MaxContentLength),
    SI_contentLength INTEGER,
    SI_format CHARACTER VARYING(8),
    SI_height INTEGER,
    SI_importTime TIMESTAMP(6),
    SI_updateTime TIMESTAMP(6),
    SI_width INTEGER
  )
  INSTANTIABLE
  NOT FINAL

METHOD SI_StillImage
  (content BINARY LARGE OBJECT(SI_MaxContentLength),
   comment CHARACTER VARYING(SI_MaxCommentLength))
  RETURNS SI_StillImage
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

METHOD SI_StillImage
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS SI_StillImage
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

METHOD SI_setContent
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS SI_StillImage
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

METHOD SI_commentText
  (text CHARACTER VARYING(SI_MaxCommentLength))

```

```

RETURNS SI_StillImage
SELF AS RESULT
LANGUAGE SQL
DETERMINISTIC
CONTAINS SQL
CALLED ON NULL INPUT

```

****Editor's Note 5-003****

The original SI_StillImage ADT defined in SQL/MM MCI-012 contained a thumbnail attribute which was meant to contain a “smaller” image. While it was agreed to delete this attribute since SQL3 does not yet support “deep references” it was suggested that a thumbnail could be supplied for some or all subtypes as a virtual attribute (computed from *content* attribute).

Definitional Rules

- 1) *SI_MaxCommentLength* is the implementation-defined maximum length for the character representation of the *SI_StillImage* attribute *SI_comment*.
- 2) *SI_MaxContentLength* is the implementation-defined maximum length for the binary representation of the *SI_StillImage* attribute *SI_content*.

Description

- 1) The *SI_StillImage* type provides for public use:
 - a) a method *SI_StillImage*(*BINARY LARGE OBJECT*, *CHARACTER VARYING*),
 - b) a method *SI_StillImage*(*BINARY LARGE OBJECT*),
 - c) a method *SI_setContent*(*BINARY LARGE OBJECT*),
 - d) a method *SI_commentText*(*CHARACTER VARYING*).
- 2) The attributes *SI_bitsPerColor*, *SI_bitsPerPixel*, *SI_comment*, *SI_content*, *SI_contentLength*, *SI_format*, *SI_height*, *SI_importTime*, *SI_updateTime* and *SI_width* are read-only. There are no GRANT statements granting EXECUTE privilege on the mutator methods for *SI_bitsPerColor*, *SI_bitsPerPixel*, *SI_comment*, *SI_content*, *SI_contentLength*, *SI_format*, *SI_height*, *SI_importTime*, *SI_updateTime* and *SI_width*.

****Editor's Note 5-013****

As Editor's Note 5-001 pointed out bitsPerColor and bitsPerPixel are not generally applicable to all still images. These fields should be moved to SI_Jpeg. The more general fields colorSpace and pixelType should be added. See SQL/MM SYD-009R1 CD Ballot Comment SEQ#33.

****Editor's Note 5-015****

The SI_StillImage type appears to be oriented to raster images. It needs to be generalized to cover both raster and vector images and then subtyped for these two general subtypes.

****Editor's Note 5-016****

The binary large object is the only representation for image content in the SI_StillImage type, which implies that image contents must be stored in a database. This presents less flexibility to applications that may have millions of images already resided in remote and distributed repositories. Application may choose to build a database to manage the repositories without uploading image contents into the database. Thus, it's necessary to support image objects stored in database with references to external repositories. One option for this purpose is to add *location* and *locationType* attributes in the SI_StillImage type. The following table defines several content location types that SI_StillImage can support.

LOCATION TYPE	DESCRIPTION
-----	-----
SI_BLOB	Binary large object in database
SI_SERVER	Content in server file system
SI_CLIENT	Content in client file system (local network)
SI_URL	Content in remote web site

Table x Image Location Types

5.1.2 SI_StillImage Methods

Purpose

Return a specified *SI_StillImage* value.

Definition

```
CREATE METHOD SI_StillImage
  (content BINARY LARGE OBJECT(SI_MaxContentLength),
   comment CHARACTER VARYING(SI_MaxCommentLength),
 RETURNS SI_StillImage
 FOR SI_StillImage
 RETURN SELF.
   SI_bitsPerColor(SI_bitsPerColor(content)).
   SI_bitsPerPixel(SI_bitsPerPixel(content)).
   SI_comment(comment).
   SI_content(content).
   SI_contentLength(LENGTH(content)).
   SI_format(SI_format(content)).
   SI_height(SI_height(content)).
   SI_importTime(CURRENT_TIMESTAMP).
   SI_updateTime(NULL).
   SI_width(SI_width(content))
```

```
CREATE METHOD SI_StillImage
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
 RETURNS SI_StillImage
 FOR SI_StillImage
 RETURN SELF.
   SI_bitsPerColor(SI_bitsPerColor(content)).
   SI_bitsPerPixel(SI_bitsPerPixel(content)).
   SI_comment(NULL).
   SI_content(content).
   SI_contentLength(LENGTH(content)).
   SI_format(SI_format(content)).
   SI_height(SI_height(content)).
   SI_importTime(CURRENT_TIMESTAMP).
   SI_updateTime(NULL).
   SI_width(SI_width(content))
```

Definitional Rules

- 1) *SI_MaxCommentLength* is the implementation-defined maximum length for the character representation of the *SI_StillImage* attribute *SI_comment*.
- 2) *SI_MaxContentLength* is the implementation-defined maximum length for the binary representation of the *SI_StillImage* attribute *SI_content*.

Description

- 1) The method *SI_StillImage(BINARY LARGE OBJECT, CHARACTER VARYING)* takes the following input parameters:
 - a) a *BINARY LARGE OBJECT* value *content*,
 - b) a *CHARACTER VARYING* value *comment*.
- 2) The method *SI_StillImage(BINARY LARGE OBJECT)* takes the following input parameter:
 - a) a *BINARY LARGE OBJECT* value *content*.

5.1.3 SI_setContent Method

Purpose

Update the *SI_StillImage* content.

Definition

```
CREATE METHOD SI_setContent
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS SI_StillImage
  FOR SI_StillImage
  BEGIN
    DECLARE NullInstance CONDITION FOR SQLSTATE '2202D';
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF01';

    IF SELF IS NULL THEN
      SIGNAL NullInstance
        SET MESSAGE_TEXT 'NULL image value';
    END IF;
    IF SI_format(content) <> SELF.SI_format THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'incorrect image format';
    END IF;

    RETURN SELF.
      SI_bitsPerColor(SI_bitsPerColor(content)).
      SI_bitsPerPixel(SI_bitsPerPixel(content)).
      SI_content(content).
      SI_contentLength(LENGTH(content)).
      SI_height(SI_height(content)).
      SI_updateTime(CURRENT_TIMESTAMP).
      SI_width(SI_width(content));
  END
```

Definitional Rules

- 1) *SI_MaxContentLength* is the implementation-defined maximum length for the binary representation of the *SI_StillImage* attribute *SI_content*.

Description

- 1) The method *SI_setContent*(*BINARY LARGE OBJECT*) takes the following input parameter:
 - a) a *BINARY LARGE OBJECT* value *content*.

5.1.4 SI_commentText Method

Purpose

Update an *SI_StillImage* instance from a character string comment.

Definition

```
CREATE METHOD SI_commentText
  (text CHARACTER VARYING(SI_MaxCommentLength))
  RETURNS SI_StillImage
  FOR SI_StillImage
  BEGIN
    DECLARE NullInstance CONDITION FOR SQLSTATE '2202D';

    IF SELF IS NULL THEN
      SIGNAL NullInstance
        SET MESSAGE_TEXT 'NULL image value';
    END IF;

    RETURN SELF.
      SI_comment(text).
      SI_updateTime(CURRENT_TIMESTAMP);
  END
```

Definitional Rules

- 1) *SI_MaxCommentLength* is the implementation-defined maximum length for the character representation of the *SI_StillImage* attribute *SI_comment*.

Description

- 1) The method *SI_commentText*(*CHARACTER VARYING*) takes the following input parameter:
 - a) a *CHARACTER VARYING* value *text*.

5.1.5 Private Functions

Purpose

These functions are only intended to be used within the methods *SI_StillImage*, *SI_setContent* and *SI_commentText*.

Definition

```
CREATE FUNCTION SI_bitsPerColor
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS INTEGER
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH
  BEGIN
    --
    -- !! See Description
    --
  END
```

```
CREATE FUNCTION SI_bitsPerPixel
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS INTEGER
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH
  BEGIN
    --
    -- !! See Description
    --
  END
```

```
CREATE FUNCTION SI_height
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS INTEGER
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH
  BEGIN
    --
    -- !! See Description
    --
  END
```

```
CREATE FUNCTION SI_format
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS CHARACTER VARYING(8)
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH
  BEGIN
    --
    -- !! See Description
    --
  END
```

```

CREATE FUNCTION SI_width
  (content BINARY LARGE OBJECT(SI_MaxContentLength))
  RETURNS INTEGER
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH
  BEGIN
    --
    -- !! See Description
    --
  END

```

Definitional Rules

- 1) *SI_MaxContentLength* is the implementation-defined maximum length for the binary representation of the *SI_StillImage* attribute *SI_content*.

Description

- 1) The private function *SI_bitsPerColor*(*BINARY LARGE OBJECT*) determines and returns the number of bits used to represent the true color of a pixel.
- 2) The private function *SI_bitsPerPixel*(*BINARY LARGE OBJECT*) determines and returns the maximum number of bits used to represent a pixel.
- 3) The private function *SI_height*(*BINARY LARGE OBJECT*) determines and returns the height in pixels of an image.
- 4) The private function *SI_format*(*BINARY LARGE OBJECT*) determines the image format of its argument. The result is the null value if the argument does not contain an image format that is amongst the implementation-defined set of supported image formats.
- 5) The private function *SI_width*(*BINARY LARGE OBJECT*) determines and returns the width in pixels of an image.

****Editor's Note 5-010****

SQL/MM Part 5: Still Image should provide a mechanism for an application to determine which facilities are supported. If possible this should be defined in the same fashion as any similar mechanism is defined in any other parts of SQL/MM.

****Editor's Note 5-011****

Conversion functions are required to convert instances of subtypes of *SI_StillImage* to instances of other formats in that subtype family.

6 Feature Types

The types in this family provide for the manipulation of still image features.

6.1 SI_Feature Type and Routines

6.1.1 SI_Feature Type

Purpose

The *SI_Feature* type provides the definition of the root type of the basic feature types.

Definition

```
CREATE TYPE SI_Feature
  NOT INSTANTIABLE
  NOT FINAL

  METHOD SI_Score
    (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
```

Definitional Rules

None.

Description

- 1) The *SI_Feature* type provides for public use:
 - a) a method *SI_Score(SI_StillImage)*.
- 2) *SI_Feature* values cannot be created. Only values of subtypes of *SI_Feature* can be created.

6.1.2 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_Feature* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_Feature

  RETURN 1.0
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* is a dummy method that will never be called since there are no *SI_Feature* values which are not values of a subtype of *SI_Feature*.

****Editor's Note 5-014****

The result of *SI_Score* needs to be specified more exactly in the case the feature (i.e. SELF) or the image to be scored is the NULL value. One alternative would be to designate this method as RETURNS NULL ON NULL INPUT to specify that it returns null if either SELF or the image parameter is the null value.

6.2 SI_AverageColor Type and Routines

6.2.1 SI_AverageColor Type

Purpose

Provide the definition of the feature type *SI_AverageColor* and facilities for scoring *SI_StillImage* values using values of the *SI_AverageColor* type.

Definition

```
CREATE TYPE SI_AverageColor
  UNDER SI_Feature
  AS (
    SI_AverageColorSpec SI_ColorType
  )
  INSTANTIABLE
  NOT FINAL

METHOD SI_AverageColor
  (RedValue INTEGER, GreenValue INTEGER, BlueValue INTEGER)
  RETURNS SI_AverageColor
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

  OVERRIDING METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
```

Definitional Rules

None.

Description

- 1) The *SI_AverageColor* type provides for public use:
 - a) a method *SI_AverageColor(INTEGER, INTEGER, INTEGER)*,
 - b) an overriding method *SI_Score(SI_StillImage)*,
 - c) a function *SI_findAverageColor(SI_StillImage)*.
- 2) The *SI_AverageColor* type represents average color values using the attribute:
 - a) *SI_AverageColorSpec* of type *SI_ColorType*, the attributes of which are intensity values for the base colors red, green, and blue.
- 3) The attribute *SI_AverageColorSpec* is not for public use. There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attribute *SI_AverageColorSpec*.

6.2.2 SI_AverageColor Method

Purpose

Return a specified *SI_AverageColor* value.

Definition

```
CREATE METHOD SI_AverageColor
  (RedValue INTEGER, GreenValue INTEGER, BlueValue INTEGER)
  RETURNS SI_AverageColor
  FOR SI_AverageColor
  BEGIN
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF03';

    IF RedValue IS NULL OR
       RedValue < 0 OR
       RedValue > SI_MaxAverageValue OR
       GreenValue IS NULL OR
       GreenValue < 0 OR
       GreenValue > SI_MaxAverageValue OR
       BlueValue IS NULL OR
       BlueValue < 0 OR
       BlueValue > SI_MaxAverageValue
    THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'incorrect average color feature
          specification';
    END IF;
    RETURN SELF.
      SI_AverageColorSpec(NEW SI_ColorType(RedValue, GreenValue,
        BlueValue));
  END
```

Definitional Rules

- 1) *SI_MaxAverageValue* is the implementation-defined maximum value that can be specified for any of the base colors defining an *SI_AverageColor* feature value.

Description

- 1) The method *SI_AverageColor(INTEGER, INTEGER, INTEGER)* takes the following input parameters:
 - a) an *INTEGER* value *RedValue*,
 - b) an *INTEGER* value *GreenValue*,
 - c) an *INTEGER* value *BlueValue*.
- 2) If any of the input arguments is the null value, is less than 0 (zero) or greater than *SI_MaxAverageValue*, then an exception condition is raised: *SQL/MM Still Image - incorrect average color feature specification*.

6.2.3 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_AverageColor* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_AverageColor
  BEGIN
    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* returns a value between 0 (zero) and 1 (one). The higher the returned value, the better the average color of an image is characterized by the average color represented by the *SI_AverageColor* value used for scoring that image. The exact relationship between the values of *SI_AverageColor*, *SI_StillImage* and the result of *SI_Score(SI_StillImage)* is implementation-dependent.

6.2.4 SI_findAverageColor Function

Purpose

Get the *SI_AverageColor* value from an *SI_StillImage* value.

Definition

```
CREATE FUNCTION SI_findAverageColor
  (image SI_StillImage)
  RETURNS SI_AverageColor
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH

  BEGIN
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF04';

    IF image IS NULL
    THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'null input image';
    END IF;

    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The function *SI_findAverageColor(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The function *SI_findAverageColor(SI_StillImage)* derives an *SI_AverageColor* value from the input argument *image*. For that end, the red, green, and blue contents of all pixels are added and divided by the number of pixels.
- 3) If the input argument *image* is the null value, then an exception condition is raised: *SQL/MM Still Image - null input image*.

6.3 SI_ColorHistogram Type and Routines

6.3.1 SI_ColorHistogram Type

Purpose

Provide the definition of the feature type *SI_ColorHistogram* and facilities for scoring *SI_StillImage* values using values of the *SI_ColorHistogram* type.

Definition

```
CREATE TYPE SI_ColorHistogram
  UNDER SI_Feature
  AS (
    SI_ColorsList SI_ColorType ARRAY[SI_MaxHistogramLength],
    SI_FrequenciesList
      DOUBLE PRECISION ARRAY[SI_MaxHistogramLength]
  )
  INSTANTIABLE
  NOT FINAL

METHOD SI_ColorHistogram
  (Colors      SI_ColorType ARRAY[SI_MaxHistogramLength],
   Frequencies DOUBLE PRECISION ARRAY[SI_MaxHistogramLength]
  )
  RETURNS SI_ColorHistogram
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

  OVERRIDING METHOD SI_Score
    (image SI_StillImage)
  RETURNS DOUBLE PRECISION
```

Definitional Rules

- 1) The *SI_MaxHistogramLength* is the implementation-defined maximum number of color/frequency pairs that are admissible in an *SI_ColorHistogram* feature value.

Description

- 1) The *SI_ColorHistogram* type provides for public use:
 - a) a method *SI_ColorHistogram(SI_ColorType ARRAY, DOUBLE PRECISION ARRAY)*,
 - b) an overriding method *SI_Score(SI_StillImage)*,
 - c) a function *SI_findColorHistogram(SI_StillImage)*.

- 2) The *SI_ColorHistogram* type represents a list of color/frequencies pairs using the attributes:
 - a) *SI_ColorsList*, an array of *SI_ColorType* values,
 - b) *SI_FrequenciesList*, an array of *DOUBLE PRECISION* values ranging from 0 (zero) to 100. The *i*-th value in this array determines the relative frequency of the *i*-th color value in *SI_ColorsList*. *SI_ColorsList* and *SI_FrequenciesList* have the same number of elements.
- 3) There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attributes *SI_ColorsList* and *SI_FrequenciesList*.

6.3.2 SI_ColorHistogram Method

Purpose

Return a specified *SI_ColorHistogram* value.

Definition

```

CREATE METHOD SI_ColorHistogram
  (Colors      SI_ColorType ARRAY[SI_MaxHistogramLength],
   Frequencies DOUBLE PRECISION ARRAY[SI_MaxHistogramLength]
  )
RETURNS SI_ColorHistogram
FOR SI_ColorHistogram
BEGIN
  DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF05';

  DECLARE i INTEGER;

  IF Colors      IS NULL OR
     Frequencies IS NULL OR
     CARDINALITY(Colors) <> CARDINALITY(Frequencies)
  THEN
    SIGNAL InvalidInput
      SET MESSAGE_TEXT 'incorrect color histogram feature
        specification';
  END IF;

  SET i = 1;
  WHILE i <= CARDINALITY(Frequencies) DO
    IF Frequencies[i] < 0.0 OR Frequencies[i] > 100.0 THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'incorrect color histogram feature
          specification';
    END IF;
    SET i = i + 1;
  END WHILE;

  RETURN SELF.
    SI_ColorsList(Colors).
    SI_FrequenciesList(Frequencies);
END

```

Definitional Rules

- 1) *SI_MaxHistogramLength* is the implementation-defined maximum number of color/frequency pairs that are admissible in an *SI_ColorHistogram* feature value.

Description

- 1) The function *SI_ColorHistogram*(*SI_ColorType* ARRAY, *DOUBLE PRECISION* ARRAY) takes the following input parameters:
 - a) an *SI_ColorType* ARRAY value *Colors*,
 - b) a *DOUBLE PRECISION* ARRAY value *Frequencies*.
- 2) If any of the input arguments is the null value, or if a frequency value is less than 0 (zero) or greater than 100, then an exception condition is raised: *SQL/MM Still Image - incorrect color histogram feature specification*.

6.3.3 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_ColorHistogram* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_ColorHistogram
  BEGIN
    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameters:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* returns a value between 0 (zero) and 1 (one). The higher the returned value, the better the color histogram of an image is characterized by the color histogram represented by the *SI_ColorHistogram* value used for scoring that image. The exact relationship between the values of *SI_ColorHistogram*, *SI_StillImage*, and the result of *SI_Score(SI_StillImage)* is implementation-dependent.

6.3.4 SI_findColorHistogram Function

Purpose

Get the *SI_ColorHistogram* value from an *SI_StillImage* value.

Definition

```
CREATE FUNCTION SI_findColorHistogram
  (image SI_StillImage)
  RETURNS SI_ColorHistogram
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH

BEGIN
  DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF04';

  IF image IS NULL
  THEN
    SIGNAL InvalidInput
      SET MESSAGE_TEXT 'null input image';
  END IF;

  --
  -- !! See Description
  --
END
```

Definitional Rules

None.

Description

- 1) The function *SI_findColorHistogram(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The function *SI_findColorHistogram(SI_StillImage)* derives an *SI_ColorHistogram* value from the input argument *image*. For that end, a relative frequency count of the triples (red, green, blue) is obtained from all pixels of *image*.
- 3) If the input argument *image* is the null value, then an exception condition is raised: *SQL/MM Still Image - null input image*.

6.4 *SI_PositionalColor* Type and Routines

6.4.1 *SI_PositionalColor* Type

Purpose

Provide the definition of the feature type *SI_PositionalColor* and facilities for scoring *SI_StillImage* values using values of the *SI_PositionalColor* type.

Definition

```
CREATE TYPE SI_PositionalColor
  UNDER SI_Feature
  AS (
    SI_AverageColorPositions
    SI_AverageColor ARRAY[SI_NumberSections]
  )
  INSTANTIABLE
  NOT FINAL

  OVERRIDING METHOD SI_Score
    (image SI_StillImage)
  RETURNS DOUBLE PRECISION
```

Definitional Rules

- 1) The *SI_NumberSections* is the implementation-defined number of *SI_AverageColor* values that are represented by an *SI_PositionalColor* value.

Description

- 1) The *SI_PositionalColor* type provides for public use:
 - a) an overriding method *SI_Score(SI_StillImage)*,
 - b) a function *SI_findPositionalColor(SI_StillImage)*.
- 2) The *SI_PositionalColor* type represents lists of *SI_AverageColor* values using the attribute:
 - a) *SI_AverageColorPositions*, an array of *SI_AverageColor* elements.
- 3) There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attribute *SI_AverageColorPositions*.

6.4.2 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_PositionalColor* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_PositionalColor
  BEGIN
    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* returns a value between 0 (zero) and 1 (one). For scoring an image, that image is effectively divided into *n* by *m* sections, such that the product of *n* and *m* equals *SI_NumberSections*. The higher the returned value, the better the *n* by *m* average colors of an image is characterized by the average colors represented by the *SI_PositionalColor* value used for scoring that image. The exact relationship between the values of *SI_PositionalColor*, *SI_StillImage*, and the result of *SI_Score(SI_StillImage)* is implementation-dependent.

6.4.3 SI_findPositionalColor Function

Purpose

Get the *SI_PositionalColor* value from an *SI_StillImage* value.

Definition

```
CREATE FUNCTION SI_findPositionalColor
  (image SI_StillImage)
  RETURNS SI_PositionalColor
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH

  BEGIN
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF04';

    IF image IS NULL
    THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'null input image';
    END IF;

    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The function *SI_findPositionalColor(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The function *SI_findPositionalColor(SI_StillImage)* derives an *SI_PositionalColor* value from the input argument *image*. For that end, *image* is effectively divided into *n* by *m* sections, and for each section, the average color value is determined. Further details on the relationship between *image* and the resulting *SI_PositionalColor* value are implementation-dependent.
- 3) If the input argument *image* is the null value, then an exception condition is raised: *SQL/MM Still Image - null input image*.

6.5 SI_Texture Type and Routines

6.5.1 SI_Texture Type

Purpose

Provide the definition of the feature type *SI_Texture* and facilities for scoring *SI_StillImage* values using values of the *SI_Texture* type.

Definition

```
CREATE TYPE SI_Texture
  UNDER SI_Feature
  AS (
    SI_TextureEncoding CHARACTER VARYING(SI_MaxTextureLength)
  )
  INSTANTIABLE
  NOT FINAL

  OVERRIDING METHOD SI_Score
    (image SI_StillImage)
  RETURNS DOUBLE PRECISION
```

Definitional Rules

- 1) The *SI_MaxTextureLength* is the implementation-defined number of bytes needed for the implementation-defined encoded representation of an *SI_Texture*.

Description

- 1) The *SI_Texture* type provides for public use:
 - a) an overriding method *SI_Score(SI_StillImage)*,
 - b) a function *SI_findTexture(SI_StillImage)*.
- 2) The *SI_Texture* type provides for the representation of image textures using the attribute:
 - a) *SI_TextureEncoding* of type *CHARACTER VARYING*, the values of which represent image texture characteristics such as coarseness, contrast, and directionality in an implementation-dependent fashion.
- 3) There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attribute *SI_TextureEncoding*.

6.5.2 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_Texture* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_Texture
  BEGIN
    --
    -- !! See Description
    --
  END
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameters:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* returns a value between 0 (zero) and 1 (one). The exact relationship between the values of *SI_Texture*, *SI_StillImage* and the result of *SI_Score(SI_StillImage)* is implementation-dependent.

6.5.3 SI_findTexture Function

Purpose

Get the *SI_Texture* value from an *SI_StillImage* value.

Definition

```

CREATE FUNCTION SI_findTexture
  (image SI_StillImage)
  RETURNS SI_Texture
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT
  STATIC DISPATCH

BEGIN
  DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF04';

  IF image IS NULL
  THEN
    SIGNAL InvalidInput
      SET MESSAGE_TEXT 'null input image';
  END IF;

  --
  -- !! See Description
  --
END

```

Definitional Rules

None.

Description

- 1) The function *SI_findTexture(SI_StillImage)* takes the following input parameters:
 - a) an *SI_StillImage* value *image*.
- 2) The function *SI_findTexture(SI_StillImage)* derives an *SI_Texture* value from the input argument *image*. The relationship between *image* and the resulting *SI_Texture* value are implementation-dependent.
- 3) If the input argument is the null value, then an exception condition is raised: *SQL/MM Still Image - null input image*.

6.6 SI_FeatureList Type and Routines

6.6.1 SI_FeatureList Type

Purpose

Provide the definition of the type *SI_FeatureList* and facilities for scoring *SI_StillImage* values using values of the *SI_FeatureList* type.

Definition

```

CREATE TYPE SI_FeatureList
  AS (
    SI_Features SI_Feature ARRAY[SI_MaxFeatureNumber],
    SI_Weights DOUBLE PRECISION ARRAY[SI_MaxFeatureNumber]
  )
  INSTANTIABLE
  NOT FINAL

METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

METHOD SI_Append
  (feature SI_Feature, weight DOUBLE PRECISION)
  RETURNS SI_FeatureList
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT,

METHOD SI_FeatureList
  (firstFeature SI_Feature, weight DOUBLE PRECISION)
  RETURNS SI_FeatureList
  SELF AS RESULT
  LANGUAGE SQL
  DETERMINISTIC
  CONTAINS SQL
  CALLED ON NULL INPUT

```

Definitional Rules

- 1) *SI_MaxFeatureNumber* is the implementation-defined maximum number of features that can be represented in an *SI_FeatureList* value.

Description

- 1) The *SI_FeatureList* type provides for public use:
 - a) a method *SI_Score(SI_StillImage)*,
 - b) a method *SI_Append(SI_Feature, DOUBLE PRECISION)*,
 - c) a method *SI_FeatureList(SI_Feature, DOUBLE PRECISION)*.
- 2) A value of type *SI_FeatureList* represents a list of weighted *SI_Feature* values using the attributes:
 - a) *SI_Features*, an array with elements of type *SI_Feature*.
 - b) *SI_Weights*, an array with elements of type *DOUBLE PRECISION* for representing relative weights by values greater than 0 (zero). The *i*-th element in this array determines the relative weight of the *i*-th *SI_Feature* element *SI_Features*. *SI_Features* and *SI_Weights* have the same number of elements.
- 3) There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attributes *SI_Features* and *SI_Weights*.

6.6.2 SI_Score Method

Purpose

Determine how well an image value scores for a given *SI_FeatureList* value.

Definition

```
CREATE METHOD SI_Score
  (image SI_StillImage)
  RETURNS DOUBLE PRECISION
  FOR SI_FeatureList
  BEGIN
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF06';
    DECLARE i INTEGER;
    DECLARE totalWeight DOUBLE PRECISION;
    DECLARE result DOUBLE PRECISION;

    IF SELF IS NULL OR
       image IS NULL
    THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'incorrect feature list or image
          specification';
    END IF;

    SET i = 1;
    SET result = 0.0;
    SET totalWeight = 0.0;
    WHILE (i <= CARDINALITY(SELF.SI_Features)) DO
      SET result = result +
        SELF.SI_Features[i].SI_Score(image) *
        SELF.SI_Weights[i];
      SET totalWeight = totalWeight + SELF.SI_Weights[i];
      SET i = i + 1;
    END WHILE;
    RETURN result/totalWeight;
  END
```

Definitional Rules

None.

Description

- 1) The method *SI_Score(SI_StillImage)* takes the following input parameter:
 - a) an *SI_StillImage* value *image*.
- 2) The method *SI_Score(SI_StillImage)* returns a value between 0 (zero) and 1 (one). For an input argument *image*, and for *i* ranging from 1 to the number of elements of *SELF.SI_Features*, let *G* the sum of all elements of *SELF.SI_Weights*; and *S_i* the product of *SELF.SI_Features[i].SI_Score(image)* with *SELF.SI_Weights[i]* divided by *G*; then the result returned by *SI_Score(SI_StillImage)* is the sum of all *S_i*.
- 3) If SELF or the input argument of *SI_Score(SI_StillImage)* is the null value, then an exception condition is raised: *SQL/MM Still Image - incorrect feature list or image specification*.

6.6.3 SI_Append Method

Purpose

Extend an *SI_FeatureList* value by another (*SI_Feature*, *DOUBLE PRECISION*) pair.

Definition

```
CREATE METHOD SI_Append
  (feature SI_Feature, weight DOUBLE PRECISION)
  RETURNS SI_FeatureList
  FOR SI_FeatureList
  BEGIN
    DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF07';
    DECLARE i INTEGER;

    IF CARDINALITY(SELF.SI_Features) = SI_MaxFeatureNumber OR
       feature IS NULL OR
       weight IS NULL OR
       weight <= 0.0
    THEN
      SIGNAL InvalidInput
        SET MESSAGE_TEXT 'incorrect feature list specification';
    END IF;

    SET SELF.SI_Features =
      CONCATENATE(SELF.SI_Features, ARRAY[feature]);
    SET SELF.SI_Weights =
      CONCATENATE(SELF.SI_Weights, ARRAY[weight]);

    RETURN SELF;
  END
```

Definitional Rules

- 1) *SI_MaxFeatureNumber* is the implementation-defined maximum number of features that can be represented in an *SI_FeatureList* value.

Description

- 1) The method *SI_Append(SI_Feature, DOUBLE PRECISION)* takes the following input parameters:
 - a) an *SI_Feature* value *feature*,
 - b) a *DOUBLE PRECISION* value *weight*.
- 2) The method *SI_Append(SI_Feature, DOUBLE PRECISION)* appends the first and second input arguments to *SELF.SI_Features* and *SELF.SI_Weights*, respectively.
- 3) If *SELF* or one of the input arguments *feature* or *weight* is the null value, or if *weight* is not greater than 0 (zero), then an exception condition is raised: *SQL/MM Still Image - incorrect feature list specification*.

6.6.4 SI_FeatureList Method

Purpose

Return an initial (i.e. length 1 (one)) *SI_FeatureList* value from an *SI_Feature* and a *DOUBLE PRECISION* value.

Definition

```
CREATE METHOD SI_FeatureList
  (firstFeature SI_Feature, weight DOUBLE PRECISION)
  RETURNS SI_FeatureList
  FOR FI_FeatureList

BEGIN
  DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF07';
  DECLARE i INTEGER;

  IF firstFeature IS NULL OR
     weight IS NULL OR
     weight <= 0.0
  THEN
    SIGNAL InvalidInput
      SET MESSAGE_TEXT 'incorrect feature list specification';
  END IF;

  RETURN SELF.
     SI_Features(ARRAY[firstFeature]).
     SI_Weights(ARRAY[weight]);
END
```

Definitional Rules

None.

Description

- 1) The method *SI_InitFeatureList(SI_Feature, DOUBLE PRECISION)* takes the following input parameters:
 - a) an *SI_Feature* value *firstFeature*,
 - b) a *DOUBLE PRECISION* value *weight*.
- 2) If any of the input arguments is the null value, or if *weight* is less than 0 (zero), then an exception condition is raised: *SQL/MM Still Image - incorrect feature list specification*.

6.7 Auxiliary Types and Routines

6.7.1 SI_ColorType Type

Purpose

Provide the definition of the type *SI_ColorType* and facilities for constructing values of this type.

Definition

```
CREATE TYPE SI_ColorType
AS (
    SI_RedValue INTEGER,
    SI_GreenValue INTEGER,
    SI_BlueValue INTEGER
)
INSTANTIABLE
NOT FINAL

METHOD SI_ColorType
(redValue INTEGER, greenValue INTEGER, blueValue INTEGER)
RETURNS SI_ColorType
SELF AS RESULT
LANGUAGE SQL
DETERMINISTIC
CONTAINS SQL
CALLED ON NULL INPUT
```

Definitional Rules

None.

Description

- 1) The *SI_ColorType* type provides for public use:
 - a) a method *SI_ColorType(INTEGER, INTEGER, INTEGER)*.
- 2) The *SI_ColorType* type represents color values using the attributes:
 - a) *SI_RedValue* of type *INTEGER*,
 - b) *SI_GreenValue* of type *INTEGER*,
 - c) *SI_BlueValue* of type *INTEGER*.
- 3) There are no GRANT statements granting EXECUTE privileges on the observer and mutator methods for the attributes *SI_RedValue*, *SI_GreenValue*, and *SI_BlueValue*.

6.7.2 SI_ColorType Method

Purpose

Return a specified SI_ColorType value.

Definition

```
CREATE METHOD SI_ColorType
  (redValue INTEGER, greenValue INTEGER, blueValue INTEGER)
  RETURNS SI_ColorType
  FOR SI_ColorType

BEGIN
  DECLARE InvalidInput CONDITION FOR SQLSTATE 'XXF07';

  IF redValue IS NULL OR
     greenValue IS NULL OR
     blueValue IS NULL OR
     redValue < 0 OR redValue > SI_MaxColor OR
     greenValue < 0 OR greenValue > SI_MaxColor OR
     blueValue < 0 OR blueValue > SI_MaxColor
  THEN
    SIGNAL InvalidInput
      SET MESSAGE_TEXT 'incorrect color specification';
  END IF;

  RETURN SELF.
    SI_RedValue(redValue).
    SI_GreenValue(greenValue).
    SI_BlueValue(blueValue);
END
```

Definitional Rules

- 1) *SI_MaxColor* is the implementation-defined maximum of a color value that can be represented in an *SI_ColorType* value.

Description

- 1) The method *SI_ColorType(redValue, greenValue, blueValue)* takes the following input parameters:
 - a) an *INTEGER* value *redValue*,
 - b) an *INTEGER* value *greenValue*,
 - c) an *INTEGER* value *blueValue*.
- 2) If any of the input arguments is the null value, is less than 0 (zero) or greater than *SI_MaxColor*, then an exception condition is raised: *SQL/MM Still Image - incorrect color specification*.

7 Status Codes

The character string value returned in an SQLSTATE parameter comprises a 2-character class value followed by a 3-character subclass value. The class value for each condition and the subclass value or values for each class value are specified in Table 1 - SQLSTATE class and subclass values.

The "Category" column has the following meanings: "S" means that the class value given corresponds to successful completion and is a completion condition; "W" means that the class value given corresponds to a successful completion but with a warning and is a completion condition; "N" means that the class value corresponds to a no-data situation and is a completion condition; "X" means that the class value given corresponds to an exception condition.

For a successful completion code but with a warning, the first two characters of the SQLSTATE are equal to the SQLSTATE condition code class value for *warning* (defined in Subclause 22.1, "SQLSTATE" in Part 2 of ISO/IEC 9075) and third character of the SQLSTATE is 'H'.

Table 1 – SQLSTATE class and subclass values

Category	Condition	Class	Subcondition	Subclass
X	data exception	22	null instance used in mutator method	02D
X	SQL/MM Still Image exception	XX	invalid input	F01
X	SQL/MM Still Image exception	XX	invalid image format	F02
X	SQL/MM Still Image exception	XX	incorrect average color feature specification	F03
X	SQL/MM Still Image exception	XX	null input image	F04
X	SQL/MM Still Image exception	XX	incorrect color histogram feature specification	F05
X	SQL/MM Still Image exception	XX	incorrect feature list or image specification	F06
X	SQL/MM Still Image exception	XX	incorrect color specification	F07

XX If the routine is implemented as an SQL-invoked routine, then the first two characters of the SQLSTATE are equal to the SQLSTATE condition code class for *SQL routine exception* (defined in Subclause 22.1, "SQLSTATE" in Part 2 of ISO/IEC 9075).

If the routine is implemented as an external routine, then the first two characters of the SQLSTATE are equal to the SQLSTATE condition code class for *external routine exception* (defined in Subclause 22.1, "SQLSTATE" in Part 2 of ISO/IEC 9075).

8 Conformance

8.1 Claims of conformance

Claims of conformance to this part of ISO/IEC 13249 shall state:

Editor's Note 5-009
Claims of conformance are to be supplied.

- 1) The definitions for all elements and actions that this part of ISO/IEC 13049 specifies as implementation-defined.

Annex A (informative) Implementation-defined elements

This Annex references those features that are identified in the body of this part of ISO/IEC 13249 as implementation-defined.

The term implementation-defined is used to identify characteristics that may differ between implementations, but that shall be defined for each particular implementation.

- 1) Subclause 5.1.5, "Private Functions". Description 4) The private function *SI_format(BINARY LARGE OBJECT)* determines the image format of its argument. The result is the null value if the argument does not contain an image format that is amongst the implementation-defined set of supported image formats.

A.1 Implementation-defined Meta-variables

- 1) *SI_MaxCommentLength* is the implementation-defined maximum length for the character representation of the *SI_StillImage* attribute *SI_comment*.
- 2) *SI_MaxContentLength* is the implementation-defined maximum length for the binary representation of the *SI_StillImage* attribute *SI_content*.
- 3) *SI_MaxAverageValue* is the implementation-defined maximum value that can be specified for any of the base colors defining an *SI_AverageColor* feature value.
- 4) The *SI_MaxHistogramLength* is the implementation-defined maximum number of color/frequency pairs that are admissible in an *SI_ColorHistogram* feature value.
- 5) The *SI_NumberSections* is the implementation-defined number of *SI_AverageColor* values that are represented by an *SI_PositionalColor* value.
- 6) The *SI_MaxTextureLength* is the implementation-defined number of bytes needed for the implementation-defined encoded representation of an *SI_Texture*.
- 7) *SI_MaxFeatureNumber* is the implementation-defined maximum number of features that can be represented in an *SI_FeatureList* value.
- 8) *SI_MaxColor* is the implementation-defined maximum of a color value that can be represented in an *SI_ColorType* value.

Annex B
(informative)
Implementation-dependent elements

This Annex references those places where this part of ISO/IEC 13249 states explicitly that the actions of a conforming implementation are implementation-dependent.

The term implementation-dependent is used to identify characteristics that may differ between implementations, but that are not necessarily specified for any particular implementation.

- 1) Subclause 6.2.3, "SI_Score Method" Description 2) The exact relationship between the values of *SI_AverageColor*, *SI_StillImage* and the result of *SI_Score(SI_StillImage)* is implementation-dependent.
- 2) Subclause 6.3.3, "SI_Score Method" Description 2) The exact relationship between the values of *SI_ColorHistogram*, *SI_StillImage*, and the result of *SI_Score(SI_StillImage)* is implementation-dependent.
- 3) Subclause 6.4.2, "SI_Score Method" Description 2) The exact relationship between the values of *SI_PositionalColor*, *SI_StillImage*, and the result of *SI_Score(SI_StillImage)* is implementation-dependent.
- 4) Subclause 6.4.3, "SI_findPositionalColor Function" Description 2) Further details on the relationship between *image* and the resulting *SI_PositionalColor* value are implementation-dependent.
- 5) Subclause 6.5.1, "SI_TextureType" Description 2) a) *SI_TextureEncoding* of type *CHARACTER VARYING*, the values of which represent image texture characteristics such as coarseness, contrast, and directionality in an implementation-dependent fashion.
- 6) Subclause 6.5.2, "SI_Score Method" Description 2) The exact relationship between the values of *SI_Texture*, *SI_StillImage* and the result of *SI_Score(SI_StillImage)* is implementation-dependent.
- 7) Subclause 6.5.3, "SI_findTextureFunction" Description 2) The relationship between *image* and the resulting *SI_Texture* value are implementation-dependent.

Index

—M—

mutator method, **2**, 6, 17, 22, 27, 30, 34, 38, 41

—S—

SI_Append method, 33, **36**

SI_AverageColor method, 17, **18**

SI_AverageColor type, **17**, 18, 19, 27, 46

SI_AverageColorPositions attribute, **27**

SI_AverageColorSpec attribute, **17**

SI_bitsPerColor attribute, **5**, 6

SI_bitsPerColor function, 8, 10, **12**

SI_bitsPerPixel attribute, **5**, 6

SI_bitsPerPixel function, 8, 10, **12**

SI_BlueValue attribute, **38**

SI_ColorHistogram method, 21, **23**

SI_ColorHistogram type, **21**, 24, 25, 46

SI_ColorHistogram type, 21

SI_ColorsList attribute, **21**

SI_ColorType method, 38, **39**

SI_ColorType type, **38**, 39

SI_comment attribute, **5**, 6, 8, 11

SI_commentText method, 5, **11**, 12

SI_content attribute, **5**, 6, 8, 10, 13

SI_contentLength attribute, **5**, 6

SI_Feature type, **15**, 17, 21, 27, 30, 37

SI_FeatureList method, 33, **37**

SI_FeatureList type, **33**, 34

SI_Features attribute, **33**

SI_findAverageColor function, **20**

SI_findColorHistogram function, **26**

SI_findPositionalColor function, **29**

SI_findTexture function, **32**

SI_format attribute, **5**, 6

SI_format function, 8, **12**

SI_FrequenciesList attribute, **21**

SI_GreenValue attribute, **38**

SI_height attribute, **5**, 6

SI_height function, 8, 10, **12**

SI_importTime attribute, **5**, 6

SI_MaxAverageValue, 18, 45

SI_MaxColor, 39, 45

SI_MaxCommentLength, 5, 6, 8, 11, 45

SI_MaxContentLength, 5, 6, 8, 10, 12, 13, 45

SI_MaxFeatureNumber, 33, 34, 36, 45

SI_MaxHistogramLength, 21, 23, 24, 45

SI_MaxTextureLength, 30, 45

SI_NumberSections, 27, 28, 45

SI_PositionalColor type, **27**, 28, 46

SI_RedValue attribute, **38**

SI_Score method, 15, **16**, 17, **19**, 21, **25**, 27, **28**, 30, **31**, 33, **35**

SI_setContent method, 5, **10**, 12

SI_StillImage method, 5, **8**, 12

SI_StillImage type, **5**, 6, 8, 10, 11, 13, 17, 19, 20, 21, 25, 26, 27, 28, 29, 30, 31, 32, 33, 45, 46

SI_Texture type, **30**, 31, 32, 46

SI_TextureEncoding attribute, **30**

SI_updateTime attribute, **5**, 6

SI_Weights attribute, **33**

SI_width attribute, **5**, 6

SI_width function, 8, 10, **13**

SQLSTATE, 41

SQLSTATE 2202D, **10**, **11**, **41**

SQLSTATE XXF01, 10, **41**

SQLSTATE XXF02, **41**

SQLSTATE XXF03, 18, **41**

SQLSTATE XXF04, 20, 26, 29, 32, **41**

SQLSTATE XXF05, 23, **41**

SQLSTATE XXF06, 35, **41**

SQLSTATE XXF07, 36, 37, **39**, **41**

