PD ISO/IEC TS 13249-7:2013

BSI Standards Publication

Information technology — Database languages — SQL multimedia and application packages

Part 7: History

bsi.

...making excellence a habit.™

Document Sharing Hub

Full document details: PD ISO/IEC TS 13249-7:2013

National foreword

This Published Document is the UK implementation of ISO/IEC TS 13249-7:2013.

The UK participation in its preparation was entrusted to Technical Committee IST/40, Data management and interchange.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2013.
Published by BSI Standards Limited 2013

ISBN 978 0 580 56100 9

ICS 35.060

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 June 2013.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: PD ISO/IEC TS 13249-7:2013

TECHNICAL SPECIFICATION

ISO/IEC TS 13249-7

First edition
2013-02-15

Information technology — Database languages — SQL multimedia and application packages —

Part 7: History

Technologies de l'information — Langages de bases de données — Multimédia SQL et paquetages d'application —

Partie 7: Historique

Reference number
ISO/IEC TS 13249-7:2013(E)

Documents
Document Sharing Hub

© ISO/IEC 2013

Full document details: PD ISO/IEC TS 13249-7:2013

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2013

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub

Full document details: PD ISO/IEC TS 13249-7:2013

Contents

Page

Foreword vi
Introduction vii
1 Scope 1
2 Normative references 1
3 Terms and definitions, concepts, notations and conventions 2
3.1 Terms and definitions 2
3.1.1 Terms and definitions provided in ISO/IEC 9075-1 2
3.1.2 Terms and definitions provided in ISO/IEC 9075-2 2
3.1.3 Terms and definition provided in this Technical Specification 2
3.2 Concepts 3
3.2.1 Concepts taken from ISO/IEC 9075-1 3
3.2.2 Concepts taken from ISO/IEC 9075-2 4
3.2.3 Syntactic elements taken from ISO/IEC 9075-2 4
3.2.4 Other concepts 5
3.2.5 Notations 5
3.2.6 Notations provided in ISO/IEC 13249-1 5
3.2.7 Notations provided in ISO/IEC 13249-7 5
3.3 Conventions 6
4 Concepts 6
4.1 Overview 6
4.1.1 Tracked Table and History Table 6
4.1.2 Concept of Transaction Timestamp 7
4.1.3 Operations on Tracked Table 7
4.1.4 Operations on time periods 8
4.1.5 Concept of Period Normalization 11
4.2 Structure of History Table 13
4.3 Creating History Table and Storing History Row in History Table 14
4.4 Retrieving a History Table 14
4.5 Types representing history rows 14
4.5.1 HS_History type 14
4.5.2 <TableTypeIdentifier> type 17
4.6 Complementary SQL-invoked regular functions 17
4.6.1 Constructor method of the HS_History type 18
4.6.2 Methods of the HS_History type for treating a period 18
4.6.3 Methods of the <TableTypeIdentifier> type 19
4.7 The History Information Schema 19
5 History Procedures 19
5.1 HS_CreateHistory Procedure and its related Procedures 19
5.1.1 HS_CreateHistory Procedure 19
5.1.2 HS_CreateHistoryErrorCheck Procedure 20
5.1.3 HS_CreateHistoryPrivilegeCheck procedure 23
5.1.4 HS_CreateHistoryTableSequenceNumberGenerator procedure 24
5.1.5 HS_CreateHistoryTableType Procedure 25
5.1.6 HS_CreateHistoryTable Procedure 26
5.1.7 HS_CreateInsertTrigger Procedure 26
5.1.8 HS_CreateUpdateTrigger Procedure 27
5.1.9 HS_CreateDeleteTrigger Procedure 29
5.1.10 HS_CreateHistoryTableMethod Procedure 30
5.1.11 HS_CreatePNormalizeMethod Procedure 31

Documents
Document Sharing Hub

© ISO/IEC 2013 – All rights reserved

Full document details: PD ISO/IEC TS 13249-7:2013

5.1.12 HS_InitializeHistoryTable Procedure 34
5.2 HS_DropHistory Procedure and its related Procedures 35
5.2.1 HS_DropHistory Procedure 35
5.2.2 HS_DropHistoryErrorCheck Procedure 35
5.2.3 HS_DropHistoryTableTypeMethod Procedure 36
5.2.4 HS_DropHistoryTrigger Procedure 37
5.2.5 HS_DropHistoryTable Procedure 38
5.2.6 HS_DropHistoryTableType Procedure 39
5.2.7 HS_DropHistoryTableSequenceNumberGenerator procedure 40
5.3 Utility Procedures for History 41
5.3.1 Functions for extracting an identifier 41
5.3.2 HS_CreateCommaSeparatedTrackedColumnList Procedure 44
5.3.3 HS_CreateCommaSeparatedTrackedColumnAndTypeList Procedure 45
5.3.4 Functions for constructing an identifier and <IdentifierLength> 47
5.3.5 HS_GetPrimaryKeys function 50
5.3.6 HS_GetTransactionTimestamp function 51
5.3.7 HS_GetHistoryRowSetIdentifierColumns procedure 51
5.3.8 HS_CreateCommaSeparatedIdentifierColumnList procedure 54
5.3.9 HS_CreateIdentifierColumnSelfJoinCondition procedure 56
5.3.10 Functions for constructing an identifier literal 58
5.3.11 HS_CreateCommaSeparatedTrackedColumnLiteralList procedure 60
5.4 <TableNameLength> and <ColumnNameLength> 61
5.5 Schema for <TableTypeIdentifier> Type 61
5.6 <TimestampPrecision> 62

6 History Types 62
6.1 HS_History Type and Routines 62
6.1.1 HS_History Type 62
6.1.2 HS_History Method 66
6.1.3 HS_Overlaps Methods 67
6.1.4 HS_Meets Methods 70
6.1.5 HS_Precedes Methods 71
6.1.6 HS_PrecedesOrMeets Methods 72
6.1.7 HS_Succeeds Methods 73
6.1.8 HS_SucceedsOrMeets Methods 74
6.1.9 HS_Contains Methods 75
6.1.10 HS_Equals Methods 77
6.1.11 HS_MonthInterval Method 78
6.1.12 HS_DayInterval Method 79
6.1.13 HS_Intersect Methods 80
6.1.14 HS_Union Methods 82
6.1.15 HS_Except Methods 83
6.2 <TableTypeIdentifier> Type and Routines 85
6.2.1 <TableTypeIdentifier> Type 85
6.2.2 HS_HistoryTable Method 86
6.2.3 HS_PNormalize Methods 87

7 SQL/MM History Information Schema 92
7.1 Introduction 92
7.2 HS_TRACKED_TABLES view 93
7.3 HS_TRACKED_COLUMNS view 93

8 SQL/MM History Definition Schema 93
8.1 Introduction 93
8.2 HS_TRACKED_TABLES base table 93
8.3 HS_TRACKED_COLUMNS base table 94

9 Status Codes 94

10 Conformance 96
10.1 Requirements for conformance 96
10.2 Features of ISO/IEC 9075 required in this Technical Specification 96

Documents
Document Sharing Hub

© ISO/IEC 2013 – All rights reserved

Full document details: PD ISO/IEC TS 13249-7:2013

10.3 Claims of conformance **97**

Annex A (informative) **Example Application** **98**
A.1 Introduction **98**
A.2 Storing History Rows **98**
A.3 Example of Queries to History Table **102**

Bibliography **107**

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

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 technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. 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.

In other circumstances, particularly when there is an urgent market requirement for such documents, the joint technical committee may decide to publish an ISO/IEC Technical Specification (ISO/IEC TS), which represents an agreement between the members of the joint technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/IEC TS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/IEC TS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 13249-7 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

ISO/IEC 13249 consists of the following parts, under the general title *Information technology — Database languages — SQL multimedia and application packages*:

— *Part 1: Framework*

— *Part 2: Full-Text*

— *Part 3: Spatial*

— *Part 5: Still image*

— *Part 6: Data mining*

— *Part 7: History* [Technical Specification]

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

Introduction

The purpose of ISO/IEC 13249 is to define multimedia and application specific types and their associated routines using the user-defined features in ISO/IEC 9075.

ISO/IEC 13249 is based on the content of ISO/IEC International Standard Database Language (SQL).

The organization of this Technical Specification is as follows:

1) Clause 1, "Scope", specifies the scope of this Technical Specification.

2) Clause 2, "Normative references", identifies additional standards that, through reference in this Technical Specification, constitute provisions of this Technical Specification.

3) Clause 3, "Terms, definitions, notations, and conventions", defines the definitions, concepts, notations and conventions used in this Technical Specification.

4) Clause 4, "Concepts", presents concepts used in the definition of this Technical Specification.

5) Clause 5, "History Procedures", defines the history associated routines.

6) Clause 6, "History Types", defines the user-defined types provided for the manipulation of history.

7) Clause 7, "SQL/MM History Information Schema" defines the SQL/MM History Information Schema.

8) Clause 8, "SQL/MM History Definition Schema" defines the SQL/MM History Definition Schema.

9) Clause 9, "Status Codes", defines the SQLSTATE codes used in this Technical Specification.

10) Clause 10, "Conformance", defines the criteria for conformance to this Technical Specification.

In the text of this Technical Specification, clauses begin a new page. Any resulting blank space is not significant.

The history user-defined types and routines defined in this Technical Specification adhere to the following:

a) A history user-defined type and routine are generic to history data handling. History user-defined types and routines provide the means to record changes to the rows of a persistent base table in an SQL database, so that applications using such a persistent base table shall be completely independent of whether there is any recording of changes. This means that, when changes are to be recorded, an application does not need to be modified and its behaviour remains the same.

b) History user-defined types and routines provide the means to query the recorded changes for such a table.

c) A history user-defined type does not redefine the database language SQL directly or in combination with another history data type.

The scope of this Technical Specification is limited to support for history when there are no changes to the definition of the tracked columns of a tracked table. The following operations are not supported in this Technical Specification:

a) DROP COLUMN operation to a tracked column of a tracked table.

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

b) ALTER COLUMN operation to a tracked column of a tracked table except changes of the default value.

The scope of this Technical Specification is limited to support for history when a tracked table has at least one unique constraint with NOT NULL that is not modified by any ALTER TABLE statements.

If a transaction does not have an isolation level that is SERIALIZABLE, the results in the recorded history are implementation-dependent.

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

Information technology — Database languages — SQL multimedia and application packages —

Part 7: History

1 Scope

The ISO/IEC 13249 series defines a number of packages of generic data types common to various kinds of data used in multimedia and application areas, to enable that data to be stored and manipulated in an SQL database.

This Technical Specification:

a) defines concepts specific to this Technical Specification;

b) defines history user-defined types and their associated routines.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 9075-1:2008, *Information technology — Database languages — SQL — Part 1: Framework (SQL/Framework)*

ISO/IEC 9075-2:2008, *Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation)*

ISO/IEC 9075-4:2008, *Information technology — Database languages — SQL — Part 4: Persistent Stored Modules (SQL/PSM)*

ISO/IEC 9075-11:2008, *Information technology — Database languages — SQL — Part 11: Information and Definition Schemas (SQL/Schemata)*

ISO/IEC 13249-1:2007, *Information technology — Database languages — SQL multimedia and application packages — Part 1: Framework*

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

3 Terms and definitions, concepts, notations and conventions

3.1 Terms and definitions

3.1.1 Terms and definitions provided in ISO/IEC 9075-1

This Technical Specification makes use of the following terms defined in ISO/IEC 9075-1:

a) atomic

b) fully qualified of a name of some SQL object

c) identify

d) object (as in 'x object')

e) persistent

f) SQL-session

3.1.2 Terms and definitions provided in ISO/IEC 9075-2

This Technical Specification makes us of the following terms defined in ISO/IEC 9075-2:

a) distinct (of a pair of comparable values)

b) qual (of a pair of comparable values)

c) dentical (of a pair of values)

d) SQL parameter

e) structured type

f) variable-length

3.1.3 Terms and definition provided in this Technical Specification

For the purposes of this document, the terms and definitions given in ISO/IEC 13249-1 and the following apply:

3.1.3.1
contiguous periods
sequence of two or more periods, such that, for all 1<i<=n (where n is the number of periods), the begin time of the i-th period is greater than the begin time of the (i-1)-th period and is equal to the end time of the (i-1)-th period

3.1.3.2
history row
row in a history table

3.1.3.3
history row set
set of rows in a history table that represent all the changes to a single row of a tracked table

3.1.3.4
history table
table that represents values of the tracked columns of a tracked table and the period during which all the values in each row were present in the tracked table

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

3.1.3.5
identifier column (of a tracked table)
either the primary key or one of the set of columns defined as UNIQUE and NOT NULL

3.1.3.6
period
duration of time with a begin time and an end time

NOTE In this Technical Specification a period value is a half-open duration that includes the begin time but not the end time.

3.1.3.7
period normalization
operation that makes one or more contiguous periods into a single period

3.1.3.8
period-normalized table
table resulting from selecting one or more columns in a history table, including the column(s) corresponding to the unique constraint columns with NOT NULL of the tracked table, and applying period normalization to rows that are otherwise not distinct

NOTE Each row in a period-normalized table is formed from one or more rows of a history table with the same values in one or more specified columns that relate to a continuous period, which may either be a single period from one row or contiguous periods from many rows.

3.1.3.9
tracked column
column of a tracked table for which changes are to be recorded

NOTE The tracked columns of a tracked table shall include unique constraint columns with NOT NULL of that table.

3.1.3.10
tracked row
row in a tracked table

3.1.3.11
tracked table
persistent base table for which changes are to be recorded for one or more tracked columns

3.1.3.12
transaction timestamp
timestamp value that is within the duration of an SQL-transaction

NOTE This value is implementation-dependent, preferably corresponding to the end of an SQL-transaction

3.2 Concepts

3.2.1 Concepts taken from ISO/IEC 9075-1

This Technical Specification makes use of the following concepts defined in ISO/IEC 9075-1:

a) assertion

b) domain

c) primary key

d) query

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

e) role

f) SQL-client module

g) SQL-schema

h) SQL-server module

i) SQL-transaction

j) SQLSTATE

k) trigger

l) unique constraint

3.2.2 Concepts taken from ISO/IEC 9075-2

This Technical Specification makes use of the following concepts defined in ISO/IEC 9075-2:

a) applicable role

b) authorization identifier

c) default unqualified schema name

d) default catalog name

e) enabled authorization identifier

f) parameter

g) procedure

h) sequence generator

i) SQL-path

j) SQL-session context

k) transaction timestamp

3.2.3 Syntactic elements taken from ISO/IEC 9075-2

This Technical Specification makes use of the following syntactic elements (BNF non-terminal symbols) defined in ISO/IEC 9075-2:

a) <catalog name>

b) <column name>

c) <comma>

d) <constraint name>

e) <data type or domain name>

f) <delimited identifier body>

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

g) <delimited identifier>

h) <double quote>

i) <equals operator>

j) <identifier body>

k) <local or schema qualifier>

l) <qualified identifier>

m) <quote>

n) <regular identifier>

o) <rollback statement>

p) <schema definition>

q) <schema name list>

r) <schema name>

s) <space>

t) <table name>

u) <Unicode delimiter body>

v) <Unicode delimiter identifier>

w) <unqualified schema name>

3.2.4 Other concepts

For the purposes of this document, the concepts given in ISO/IEC 13249-1 apply.

3.2.5 Notations

3.2.6 Notations provided in ISO/IEC 13249-1

For the purposes of this document, the notations given in ISO/IEC 13249-1 apply.

3.2.7 Notations provided in ISO/IEC 13249-7

This Technical Specification uses the prefix 'HS_' for view, base table, user-defined type, attribute and SQL-invoked routine names.

This Technical Specification uses the following representation in a figure for a table that includes the column of HS_Hist of the structured type, HS_History.

<column name>	*<column name>*	*⋯*	*HS_Hist (HS_BeginTime, HS_EndTime)*
Column Value	*Column Value*	*⋯*	*(Attribute Value, Attribute Value)*
Column Value	*Column Value*	*⋯*	*(Attribute Value, Attribute Value)*

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

…	…	…	…

3.3 Conventions

For the purposes of this document, the conventions given in ISO/IEC 9075-4, ISO/IEC 9075-11 and ISO/IEC 13249-1 apply.

4 Concepts

4.1 Overview

This Technical Specification provides user-defined types and routines that enable a user to specify columns of a table to record all changes to these columns and to query recorded changes. The recorded changes include tracking information on inserts, deletes, and updates of those columns of the table.

This part does not specify the means by which recorded changes are maintained. However, the concept of a history table is introduced in this clause, and it has two roles. A history table allows the precise specification of the user-defined types and routines providing the required capabilities, and it determines the way in which the recorded changes are materialised for querying.

4.1.1 Tracked Table and History Table

A tracked table is a persistent base table for which any changes to the current values of specified tracked columns are to be recorded. A history table is a means of virtualising the recording of these changes, even though there is no requirement for it to exist as a persistent base table.

A history table consists of the column of a sequence number, the columns corresponding to all tracked columns of the tracked table and the column of the structured type for the period of a history row.

Example 1:

Tracked Table TT1

ID	Column_A	Column_B	Column_C

History Table for Tracked Table TT1

HS_SEQ	ID	Column_A	Column_B	HS_Hist (HS_BeginTime, HS_EndTime)

Example 2:

Tracked Table TT2

ID	Column_A	Column_B	Column_C	Column_D	Column_E	Column_F	Column_G

History Table for Tracked Table TT2

HS_SEQ	ID	Column_A	Column_B	Column_C	Column_D	HS_Hist (HS_BeginTime, HS_EndTime)

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

In example 1, the column ID is the unique constraint column of the tracked table TT1. The columns ID, Column_A and Column_B are tracked columns of the tracked table TT1.

In example 2, the column ID is the unique constraint column of the tracked table TT2. The columns ID, Column_A, Column_B, Column_C and Column_D are tracked columns of the tracked table TT2.

The value of a begin time or an end time is automatically set by the system when an insert, update, or delete operation is executed on the tracked table TT1 or the tracked table TT2.

A row in the history table, namely history row, represents the values of the columns (in example 1, ID, Column_A and Column_B, and in example 2, ID, Column_A, Column_B, Column_C and Column_D) that existed from begin time to end time. Once a begin/end time is set to a certain non-NULL timestamp value, the value will not be changed.

The value of the column HS_SEQ is a number which is generated by an external sequence generator that is defined as START WITH 1 INCREMENT BY 1. The value of the column HS_SEQ is set when a history row is inserted into a history table.

4.1.2 Concept of Transaction Timestamp

When an insert operation, an update operation or a delete operation is executed on a tracked table, the values of the tracked columns need to be recorded along with a begin time and possibly an end time for the history period. But if multiple DML operations are executed in the same SQL-transaction, these operations will occur at different times and so using CURRENT_TIMESTAMP the resulted timestamp values may vary and would not allow the multiple changes for a transaction to be related. This is an especially serious problem if more than one table is being tracked. Thus a single time for all changes within a transaction is required. This requirement is provided by the implementation-dependent transaction timestamp for an SQL-transaction.

The value of transaction timestamp is used as the value of a begin time or an end time of a history row (see Section 4.1.3 Operations on Tracked Table).

The value of a transaction timestamp is determined by invoking the HS_GetTransactionTimestamp function defined in Subclause 5.3.9.

4.1.3 Operations on Tracked Table

A history table is created automatically for a tracked table by invoking the HS_CreateHistory procedure provided in this Technical Specification.

When the HS_CreateHistory procedure is invoked, a history table is initialized as follows. A history table is created and all rows in the tracked table are inserted into the history table, and the begin times of all history rows are set to the value of transaction timestamp of the execution time of the HS_CreateHistory procedure.

When an insert operation on a tracked table is executed, a history row, which corresponds to the inserted row in the tracked table, is inserted into the history table, and the begin time is set to the value of transaction timestamp of the transaction that executes the insert operation.

When an update operation on a tracked table is executed, for each row that is changed:

1) the end time of the row which has the latest begin time in the history row set for that row is set to the value of transaction timestamp of the transaction that executes the update operation, and

2) a history row which has the new values of the tracked columns for that row and the value of the transaction timestamp as its begin time is inserted into the history table.

When a delete operation on a tracked table is executed, for each row that is deleted, the end time of the row which has the latest begin time in the history row set for that row is set to the value of transaction timestamp of the transaction that executes the delete operation.

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

4.1.4 Operations on time periods

In order to query a history table, an application needs a means to specify conditions for the begin time or the end time when it is interested in. Also an application needs a means to extract the value of the begin time or the end time or to execute time-related operations.

Predicates to specify time-related condition to the history table are as follows. *X* and *Y* represent half-open periods ; [*b1*:*e1*) and [*b2*:*e2*) respectively, where the left square bracket shows that the begin of period is inclusive and the right parenthesis shows that the end of period is not inclusive, and *b1* and *b2* are the begin times and *e1* and *e2* are the end times.

— *X* Overlaps *Y* is:

  — true if and only if *b1* < *e2* and *b2* < *e1* are both true.

  — false if and only if either of *b1* < *e2* or *b2* < *e1* is false.

— *X* Contains *Y* is:

  — true if and only if *b1* <= *b2* and *e2* <= *e1* are both true.

  — false if and only if either of *b1* <= *b2* or *e2* <= *e1* is false.

— *X* Meets *Y* is:

  — true if and only if either of *e1* = *b2* or *e2* = *b1* is true.

  — false if and only if *e1* = *b2* and *e2* = *b1* are both false.

— *X* Precedes *Y* is:

  — true if and only if *e1* < *b2* is true.

  — false if and only if *e1* < *b2* is false.

— *X* PrecedesOrMeets Y is:

  — true if and only if either of *X* Precedes *Y* or *e1* = *b2* is true.

  — false if and only if *X* Precedes *Y* and *e1* = *b2* are both false.

— *X* Equals *Y* is:

  — true if and only if *b1* = *b2* and *e1* = *e2* are both true.

  — false if and only if either of *b1* = *b2* or *e1* = *e2* is false.

— *X* Succeeds *Y* is:

  — true if and only if *e2* < *b1* is true.

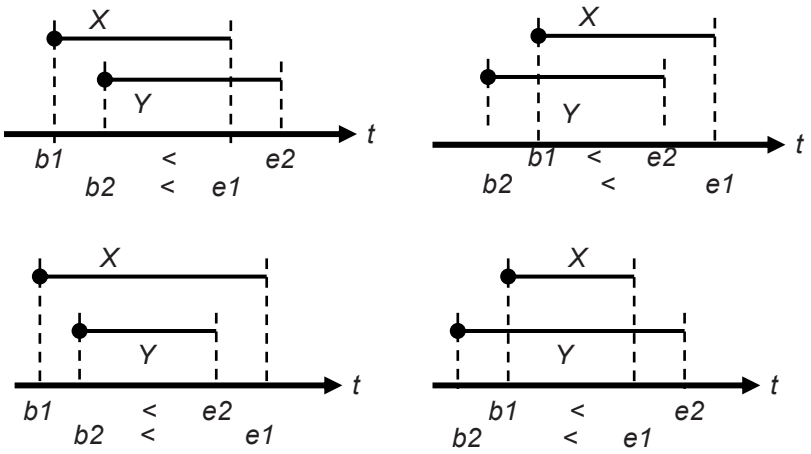
  — false if and only if *e2* < *b1* is false.

— *X* SucceedsOrMeets Y is:

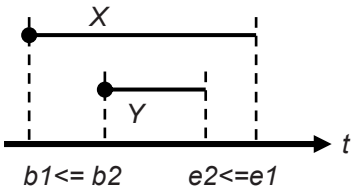
  — true if and only if either of *X* Succeeds *Y* or *e2* = *b1* is true.

  — false if and only if *X* Succeeds *Y* and *e2* = *b1* are both false.

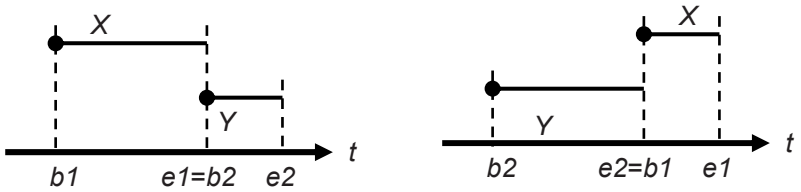
© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013



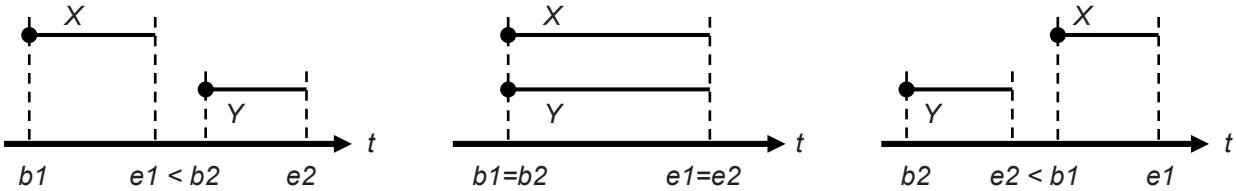
X Overlaps *Y*



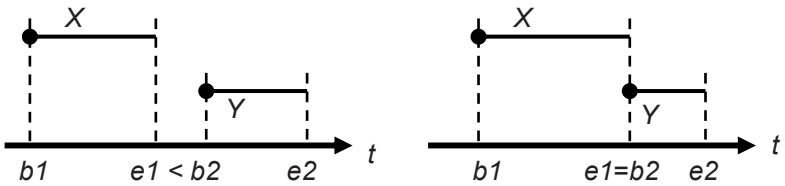
X Contains *Y*



X Meets *Y*



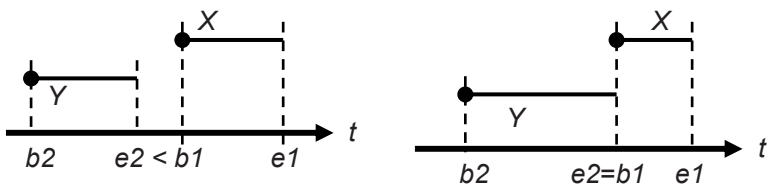
X Precedes *Y* *X* Equals *Y* *X* Succeeds *Y*



X PrecedesOrMeets *Y*

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013



X SucceedsOrMeets *Y*

Operations to obtain a time interval of a period are as follows:

- year-month interval of X is:
  - (e1 - b1) as a year-month interval.
- day-time interval of X is:
  - (e1 - b1) as a day-time interval.

Set operations on two periods are as follows.

— X Intersect Y returns:

  — [MAX(b1, b2):MIN(e1, e2)) if X Overlaps Y is true.

— X Union Y returns:

  — [MIN(b1, b2):MAX(e1, e2)) if either of X Overlaps Y or X Meets Y is true.

— X Except Y returns:

  — [b1:MIN(b2, e1)) if b1 < b2 and e1 <= e2 are both true.

  — [MAX(e2, b1):e1) if b2 <= b1 and e2 < e1 are both true.

In this Technical Specification, the following user-defined types and methods are provided to specify time-related queries to the history table.

1) User-defined types:

   a) HS_History type,

   b) <TableTypeIdentifier> type.

2) Methods:

   a) HS_Overlaps

   b) HS_Meets

   c) HS_Precedes

   d) HS_PrecedesOrMeets

   e) HS_Succeeds

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

f) HS_SucceedsOrMeets

g) HS_Contains

h) HS_Equals

i) HS_MonthInterval

j) HS_DayInterval

k) HS_Intersect

l) HS_Union

m) HS_Except

n) HS_PNormalize

4.1.5 Concept of Period Normalization

Period normalization is introduced to provide an aggregate operation for a number of history rows of a history table.

As an example, consider the case where a history table has two tracked columns in addition to the primary key, ID. In this case a history row is inserted into the history table whenever at least one of tracked columns is changed.

Now let us consider an application that requests a history table on only one of the two tracked columns that were originally specified. Since it was possible that tracked rows were added to the originally specified history table for changes in not only the specified (tracked) column but also the other tracked column, the application would have to specify a complicated query to determine the actual periods when the values of the requested column were not changed.

NOTE Some example queries of period normalization are shown in Annex A.3.

Example 1:

History Table for Tracked Table TT1

HS_SEQ	ID	Column_A	Column_B	HS_Hist (HS_BeginTime, HS_EndTime)
1	1	A1	B1	(T1, T2)
2	1	A1	B2	(T2, T3)
3	1	A2	B2	(T3, T4)
4	1	A2	B3	(T4, T5)
5	1	A3	B3	(T5, T6)
6	1	A4	B3	(T6, T7)
7	1	A4	B4	(T7, NULL)

When period normalization of Column_B is executed, as the value of the Column_B is B1 from T1 to T2, B2 from T2 to T4, B3 from T4 to T7, and B4 from T7 to now, the period-normalized table is as follows:

HS_SEQ	ID	Column_B	HS_Hist (HS_BeginTime, HS_EndTime)
1	1	B1	(T1, T2)
2	1	B2	(T2, T4)

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

4	1	B3	(T4, T7)
7	1	B4	(T7, NULL)

Example 2:

History Table for Tracked Table TT2

HS_SEQ	ID	Column_A	Column_B	Column_C	Column_D	HS_Hist (HS_BeginTime, HS_EndTime)
1	1	A1	B1	C1	D1	(T1, T3)
2	2	A1	B1	C1	D1	(T2, T4)
3	1	A2	B1	C1	D1	(T3, T5)
4	2	A2	B1	C1	D1	(T4, T6)
5	1	A2	B2	C1	D1	(T5, T7)
6	2	A2	B2	C1	D1	(T6, T8)
7	1	A2	B2	C2	D2	(T7, T9)
8	2	A2	B2	C2	D2	(T8, T10)
9	1	A3	B2	C2	D3	(T9, T11)
10	2	A3	B2	C2	D3	(T10, T11)
11	1	A3	B3	C3	D3	(T11, T12)
12	2	A3	B3	C3	D3	(T11, T13)
13	1	A3	B4	C4	D3	(T12, NULL)
14	2	A3	B4	C4	D3	(T13, NULL)

When period normalization of Column_B and Column_C is executed, for the rows whose ID is 1, the values of Column_B and Column_C are B1 and C1 from T1 to T5, B2 and C1 from T5 to T7, B2 and C2 from T7 to T11, B3 and C3 from T11 to T12, and B4 and C4 from T12 to now, and for the rows whose ID is 2, the values of Column_B and Column_C are B1 and C1 from T2 to T6, B2 and C1 from T6 to T8, B2 and C2 from T8 to T11, B3 and C3 from T11 to T13, and B4 and C4 from T13 to now. The period-normalized table is as follows:

HS_SEQ	ID	Column_B	Column_C	HS_Hist (HS_BeginTime, HS_EndTime)
1	1	B1	C1	(T1, T5)
5	1	B2	C1	(T5, T7)
7	1	B2	C2	(T7, T11)
11	1	B3	C3	(T11, T12)
13	1	B4	C4	(T12, NULL)
2	2	B1	C1	(T2, T6)
6	2	B2	C1	(T6, T8)
8	2	B2	C2	(T8, T11)
12	2	B3	C3	(T11, T13)
14	2	B4	C4	(T13, NULL)

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

The operation of “period normalization” is introduced in this Technical Specification to perform the derivation of a tracked table on a subset of the original set of tracked columns of a history table.

Period normalization of selected columns of a history table is an operation that derives a single history row from one or more history rows.

In order to execute period normalization of selected columns of a history table, the history rows of a history row set are divided into sets of history rows, whose values of the selected columns are not distinct across a set of history rows which have contiguous periods.

In each such set of history rows, from the begin time of the oldest history row in the set to the end time of the latest history row in the set, the values of the selected columns are not distinct. Therefore, the period-normalized history table holds one row, which has the following values, for each set of history rows in each history row set:

1) The values of the selected columns which are common to all of the rows in the set of history rows.

2) The value of the begin time of the oldest history row in the set of history rows.

3) The value of the end time of the latest history row in the set of history rows.

4) Null values for the other tracked columns in the history row.

4.2 Structure of History Table

Whenever an insert/update operation occurs on a row of a tracked table, a row is inserted into a history table. A row of a tracked table is called a tracked row and a row inserted into a history table is called a history row. Such history rows can be grouped as rows corresponding to a certain tracked row. A multiset of history rows corresponding to a certain tracked row is called a history row set of the tracked row.

The following type is provided to represent a period associated with a history row:

— The HS_History type with the following attributes:

— HS_BeginTime: a TIMESTAMP(<*TimestampPrecision*>) value to store a begin time;

— HS_EndTime: a TIMESTAMP(<*TimestampPrecision*>) value to store an end time;

The following conditions are held between the members of a history row set of a tracked row:

— The time order of the values of the begin times for all history rows contained in a history row set is the same as the order that the history rows were added to the history table.

— The begin time of history row A in a history row set is equal to the end time of the history row B in the history row set that was added immediately before history row A was added.

A history row is represented by the following type which is automatically generated for every tracked table:

— The <TableTypeIdentifier> type with the following attributes:

— an attribute HS_SEQ of type <*TypeOf_HS_SEQ*> to contain the sequence number which is associated with a history row;

— one attribute for each tracked column of the tracked table;

— an attribute HS_Hist of type HS_History to contain the period which is associated with a history row.

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

4.3 Creating History Table and Storing History Row in History Table

The HS_CreateHistory procedure is provided to create a history table corresponding to the specified tracked table.

In order to keep a history table up to date with changes made to the tracked table, the following triggers are created:

— An AFTER trigger to insert a new history row into the history table when a row is inserted into the tracked table.

— An AFTER trigger to update the end time of the latest history row to the transaction timestamp value and insert a new history row into the history table when the value of a tracked column is updated.

— An AFTER trigger to update the end time of the latest history row to the transaction timestamp value when a row is deleted from the tracked table.

These triggers are automatically created for every tracked table when the HS_CreateHistory procedure is invoked.

4.4 Retrieving a History Table

To retrieve a history table, the HS_HistoryTable method or the HS_PNormalize method of <TableTypeIdentifier> type should be used. The HS_HistoryTable method returns the whole history table. The HS_PNormalize method returns the table that is the result of period normalization of specified column(s) of the history table.

4.5 Types representing history rows

4.5.1 HS_History type

4.5.1.1 Attributes of the HS_History type

The HS_History type is a representation of a period, using the following attributes:

— HS_BeginTime: the start of a period;

— HS_EndTime: the end of a period;

A value *V* of the HS_History type is as follows, where *T* is a transaction timestamp that corresponds to the SQL-transaction which the HS_History constructor method is invoked:

a) *V*.HS_BeginTime cannot be null.

b) *V*.HS_BeginTime is not greater than *T*.

c) If *V*.HS_EndTime is null, then *V* represents a period extending from *V*.HS_BeginTime (inclusive) until the present moment.

d) If *V*.HS_EndTime is not null, then *V*.HS_EndTime is not less than *V*.HS_BeginTime but less than or equal to *T*. In this case *V* represents a period extending from *V*.HS_BeginTime (inclusive) through *V*.HS_EndTime (exclusive).

4.5.1.2 Methods of the HS_History type

The type HS_History provides the following methods for public use:

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

4.5.1.2.1 Constructor method

— HS_History
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS HS_History:
Generate HS_History value which has the specified TIMESTAMP values as the begin time and the end time.

4.5.1.2.2 Methods for treating a period

— HS_Overlaps
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value overlaps with the period specified by the arguments;

— HS_Overlaps(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value overlaps with the specified period;

— HS_Meets
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value meets the period specified by the arguments;

— HS_Meets(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value meets the specified period;

— HS_Precedes
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value precedes the period specified by the arguments;

— HS_Precedes(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value precedes the specified period;

— HS_PrecedesOrMeets
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value precedes or meets the period specified by the arguments;

— HS_PrecedesOrMeets(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value precedes or meets the specified period;

— HS_Succeeds
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value succeeds the period specified by the arguments;

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

— HS_Succeeds(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value succeeds the specified period;

— HS_SucceedsOrMeets
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value succeeds or meets the period specified by the arguments;

— HS_SucceedsOrMeets(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value succeeds or meets the specified period;

— HS_Contains(timePoint TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value contains the specified TIMESTAMP value;

— HS_Contains
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value contains the period specified by the arguments;

— HS_Contains(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value contains the specified period;

— HS_Equals
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS INTEGER:
Test whether the period of an HS_History value is equal to the period specified by the arguments;

— HS_Equals(hs_hist HS_History)
RETURNS INTEGER:
Test whether the period of an HS_History value is equal to the specified period;

— HS_MonthInterval()
RETURNS INTERVAL YEAR TO MONTH:
Obtain the length of the period of an HS_History value as a year-month interval;

— HS_DayInterval()
RETURNS INTERVAL DAY TO SECOND:
Obtain the length of the period of an HS_History value as a day-time interval;

— HS_Intersect
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS HS_History:
Generate a new HS_History value with the period which is the overlap of the period of an HS_History value and the period specified by the arguments;

— HS_Intersect(hs_hist HS_History)
RETURNS HS_History:
Generate a new HS_History value with the period which is the overlap of the period of an HS_History value and the specified period;

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

— HS_Union
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS HS_History:
Generate a new HS_History value with the period which is the union of the period of an HS_History value and the period specified by the arguments;

— HS_Union(hs_hist HS_History)
RETURNS HS_History:
Generate a new HS_History value with the period which is the union of the period of an HS_History value and the specified period;

— HS_Except
(beginOfPeriod TIMESTAMP(<*TimestampPrecision*>),
endOfPeriod TIMESTAMP(<*TimestampPrecision*>))
RETURNS HS_History:
Generate a new HS_History value with the period obtained from the period of an HS_History value except for the period specified by the arguments;

— HS_Except(hs_hist HS_History)
RETURNS HS_History:
Generate a new HS_History value with the period obtained from the period of an HS_History value except for the specified period;

4.5.2 <TableTypeIdentifier> type

4.5.2.1 Attributes of the <TableTypeIdentifier> type

The <TableTypeIdentifier> type is an abstraction for attributes of a history row, using the following attributes:

— an attribute HS_SEQ of type <*TypeOf_HS_SEQ*>;

— one attribute for each of the tracked column of the tracked table;

— an attribute HS_Hist of type HS_History;

4.5.2.2 Methods of the <TableTypeIdentifier> type

The type <*TableTypeIdentifier*> provides the following methods:

— HS_HistoryTable()
RETURNS TABLE(HS_SEQ <*TypeOf_HS_SEQ*>, <*AllTrackedColumnsDef*>, HS_Hist HS_History):
Obtain whole history table;

— HS_PNormalize(targetColumn CHARACTER VARYING(<*ColumnNameLength*>))
RETURNS TABLE(HS_SEQ <*TypeOf_HS_SEQ*>, <*AllTrackedColumnsDef*>, HS_Hist HS_History):
Obtain a period-normalized table of the specified one column of a history table;

— HS_PNormalize(targetColumns CHARACTER VARYING(<*ColumnNameLength*>) ARRAY)
RETURNS TABLE(HS_SEQ <*TypeOf_HS_SEQ*>, <*AllTrackedColumnsDef*>, HS_Hist HS_History):
Obtain a period-normalized table of the specified one or more columns of a history table;

4.6 Complementary SQL-invoked regular functions

To ease conformance for implementation of this Technical Specification, each method intended for public use is complemented by an SQL-invoked regular function.

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

For each such method, the type of specified method, the method name, parameter types (if any), and the name of the corresponding SQL-invoked regular function are listed in Table 1, Table 2, and Table 3 in the following section.

4.6.1 Constructor method of the HS_History type

Table 1 — Method and function name correspondences

Type Name	Method Name	Parameter Types (if any)	Function Name
HS_History	HS_History	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_History

4.6.2 Methods of the HS_History type for treating a period

Table 2 — Method and function name correspondences

Type Name	Method Name	Parameter Types (if any)	Function Name
HS_History	HS_Overlaps	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Overlaps_TT
HS_History	HS_Overlaps	HS_History	HS_Overlaps_H
HS_History	HS_Meets	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Meets_TT
HS_History	HS_Meets	HS_History	HS_Meets_H
HS_History	HS_Precedes	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Precedes_TT
HS_History	HS_Precedes	HS_History	HS_Precedes_H
HS_History	HS_PrecedesOrMeets	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_PrecedesOrMeets_TT
HS_History	HS_PrecedesOrMeets	HS_History	HS_PrecedesOrMeets_H
HS_History	HS_Succeeds	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Succeeds_TT
HS_History	HS_Succeeds	HS_History	HS_Succeeds_H
HS_History	HS_SucceedsOrMeets	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_SucceedsOrMeets_TT
HS_History	HS_SucceedsOrMeets	HS_History	HS_SucceedsOrMeets_H
HS_History	HS_Contains	TIMESTAMP(<*TimestampPrecision*>)	HS_Contains_T
HS_History	HS_Contains	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Contains_TT
HS_History	HS_Contains	HS_History	HS_Contains_H
HS_History	HS_Equals	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Equals_TT
HS_History	HS_Equals	HS_History	HS_Equals_H
HS_History	HS_MonthInterval		HS_MonthInterval
HS_History	HS_DayInterval		HS_DayInterval
HS_History	HS_Intersect	TIMESTAMP(<*TimestampPrecision*>),	HS_Intersect_TT

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

Type Name	Method Name	Parameter Types (if any)	Function Name
		TIMESTAMP(<*TimestampPrecision*>)	
HS_History	HS_Intersect	HS_History	HS_Intersect_H
HS_History	HS_Union	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Union_TT
HS_History	HS_Union	HS_History	HS_Union_H
HS_History	HS_Except	TIMESTAMP(<*TimestampPrecision*>), TIMESTAMP(<*TimestampPrecision*>)	HS_Except_TT
HS_History	HS_Except	HS_History	HS_Except_H

4.6.3 Methods of the <TableTypeIdentifier> type

Table 3 — Method and function name correspondences

Type Name	Method Name	Parameter Types (if any)	Function Name
<*TableTypeIdentifier*>	HS_HistoryTable		HS_HistoryTable
<*TableTypeIdentifier*>	HS_PNormalize	CHARACTER VARYING(<*ColumnNameLength*>)	HS_PNormalize
<*TableTypeIdentifier*>	HS_PNormalize	CHARACTER VARYING(<*ColumnNameLength*>) ARRAY	HS_PNormalize_A

4.7 The History Information Schema

This Technical Specification prescribes an Information Schema called HS_INFORMTN_SCHEMA. It contains views for the following purposes:

— a view HS_TRACKED_TABLES, which lists the tables that have an associated history table;

— a view HS_TRACKED_COLUMNS, which lists the columns that are specified as the parameter of HS_CreateHistory procedure.

5 History Procedures

5.1 HS_CreateHistory Procedure and its related Procedures

5.1.1 HS_CreateHistory Procedure

Purpose

Create a history table for the specified tracked table, create three triggers to maintain the history table, define the type corresponding to the history table, and initialize the history table.

Definition

```
CREATE PROCEDURE HS_CreateHistory
   (IN TableName CHARACTER VARYING(<TableNameLength>),
    IN TrackedColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)
```

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

```
LANGUAGE SQL
NOT DETERMINISTIC
MODIFIES SQL DATA
SQL SECURITY INVOKER
BEGIN
   CALL HS_CreateHistoryErrorCheck(TableName, TrackedColumns);
   CALL HS_CreateHistoryPrivilegeCheck(TableName, TrackedColumns);
   CALL HS_CreateHistoryTableSequenceNumberGenerator(TableName);
   CALL HS_CreateHistoryTableType(TableName, TrackedColumns);
   CALL HS_CreateHistoryTable(TableName, TrackedColumns);
   CALL HS_CreateInsertTrigger(TableName, TrackedColumns);
   CALL HS_CreateUpdateTrigger(TableName, TrackedColumns);
   CALL HS_CreateDeleteTrigger(TableName, TrackedColumns);
   CALL HS_CreateHistoryTableMethod(TableName, TrackedColumns);
   CALL HS_CreatePNormalizeMethod(TableName, TrackedColumns);
   CALL HS_InitializeHistoryTable(TableName, TrackedColumns);
END
```

Definitional Rules

1) The SQL-session context shall be such that:

   a) The value of the current default catalog name is <*CatalogName*>.

   b) The value of the current default unqualified schema name is <*SchemaName*>.

2) The enabled authorization identifiers shall include the authorization identifier of schema whose name is <*CatalogName*> . <*SchemaName*>.

NOTE An enabled authorization identifier is defined in ISO/IEC 9075.

3) If an error occurred in one of procedures invoked in the HS_CreateHistory procedure, the HS_CreateHistory procedure is implicitly ended with a <rollback statement>.

Description

1) The *HS_CreateHistory(CHARACTER VARYING(<TableNameLength>), CHARACTER VARYING(<ColumnNameLength>) ARRAY)* takes the following input parameters:

   a) *TableName*, whose value is a character representation of a table name which conforms to the Format and Syntax Rules of <table name> in ISO/IEC 9075-2.

   b) *TrackedColumns*, each element of whose value is a character representation of a column name which conforms to the Format and Syntax Rules of <column name> in ISO/IEC 9075-2.

5.1.2 HS_CreateHistoryErrorCheck Procedure

Purpose

Check whether the input parameters of the HS_CreateHistory procedure are valid.

Definition

```
CREATE PROCEDURE HS_CreateHistoryErrorCheck
   (IN TableName CHARACTER VARYING(<TableNameLength>),
    IN TrackedColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)
   LANGUAGE SQL
```


© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

```
DETERMINISTIC
READS SQL DATA
BEGIN
   DECLARE IdentifierColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY;
   DECLARE i INTEGER;
   DECLARE j INTEGER;

   --
   -- Check if the specified TableName parameter is valid
   --
   IF TableName IS NULL THEN
      SIGNAL SQLSTATE '2FF01' SET MESSAGE_TEXT =
         'table name is a null value';
   END IF;
   --
   -- Check if the table specified exists in the SQL-database
   --
   IF NOT EXISTS(SELECT *
         FROM information_schema.tables
         WHERE TABLE_CATALOG = HS_ExtractCatalogIdentifier(TableName)
            AND TABLE_SCHEMA = HS_ExtractSchemaIdentifier(TableName)
            AND TABLE_NAME = HS_ExtractTableIdentifier(TableName))
         THEN
      SIGNAL SQLSTATE '2FF02' SET MESSAGE_TEXT =
         'table does not exist';
   END IF;
   --
   -- Check if the specified table has
   --  unique constraints with NOT NULL for all columns of the constraint.
   -- If there is no unique constraint with NOT NULL,
   --  then exception condition is raised.
   --
   CALL HS_GetHistoryRowSetIdentifierColumns(
         TableName, TrackedColumns, IdentifierColumns);
   IF NOT EXISTS(SELECT *
         FROM UNNEST(IdentifierColumns) AS CLMS(CNAME)) THEN
      SIGNAL SQLSTATE '2FF13' SET MESSAGE_TEXT =
         'tracked table has no unique constraint with NOT NULL';
   END IF;
   --
   -- Check if the history table for the specified table does not exist
   --
   IF EXISTS(SELECT *
         FROM information_schema.tables
         WHERE TABLE_CATALOG = HS_ExtractCatalogIdentifier(TableName)
            AND TABLE_SCHEMA = HS_ExtractSchemaIdentifier(TableName)
            AND TABLE_NAME = HS_ConstructTableIdentifier(TableName))
         THEN
      SIGNAL SQLSTATE '2FF03' SET MESSAGE_TEXT =
         'history table already exists';
   END IF;
   --
```

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

```
   -- Check if the specified TrackedColumns parameter is valid
   --
   IF TrackedColumns IS NULL THEN
      SIGNAL SQLSTATE '2FF04' SET MESSAGE_TEXT =
         'no tracked column is specified';
   END IF;
   IF CARDINALITY(TrackedColumns) < 1 THEN
      SIGNAL SQLSTATE '2FF04' SET MESSAGE_TEXT =
         'no tracked column is specified';
   END IF;
   --
   -- For each element of an array TrackedColumns, check if
   --   a) it is not a null value.
   --   b) it is a column name of the specified table.
   --
   SET i = 1;
   WHILE i <= CARDINALITY(TrackedColumns) DO
      IF TrackedColumns[i] IS NULL THEN
      -- Specified column name is null.
         SIGNAL SQLSTATE '2FF05' SET MESSAGE_TEXT =
            'column name is a null value';
      ELSEIF NOT EXISTS(SELECT *
            FROM information_schema.columns
            WHERE
               TABLE_CATALOG = HS_ExtractCatalogIdentifier(TableName)
               AND TABLE_SCHEMA = HS_ExtractSchemaIdentifier(TableName)
               AND TABLE_NAME = HS_ExtractTableIdentifier(TableName)
               AND column_name =
                  HS_ExtractColumnIdentifier(TrackedColumns[i])) THEN
      -- The tracked table does not have a specified column.
         SIGNAL SQLSTATE '2FF06' SET MESSAGE_TEXT =
            'column does not exist';
      END IF;
      SET i = i + 1;
   END WHILE;
   --
   -- Check if all unique constraint columns are specified
   -- as the TrackedColumns parameter
   --
   SET i = 1;
   WHILE i <= CARDINALITY(IdentifierColumns) DO
      IF NOT EXISTS(SELECT *
               FROM UNNEST(TrackedColumns) AS CLMS(CNAME)
               WHERE CNAME = IdentifierColumns[i]) THEN
      -- i-th identifier column is not specified as tracked columns.
         SIGNAL SQLSTATE '2FF14' SET MESSAGE_TEXT =
            'some of unique constraint columns are not specified as tracked
            columns';
      END IF;
      SET i = i + 1;
   END WHILE;
END
```


© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

Description

1) The *HS_CreateHistoryErrorCheck(CHARACTER VARYING(<TableNameLength>), CHARACTER VARYING(<ColumnNameLength>) ARRAY)* procedure takes the following input parameters:

   a) a CHARACTER VARYING(<*TableNameLength*>) value *TableName*,

   b) a CHARACTER VARYING(<*ColumnNameLength*>) ARRAY value *TrackedColumns*.

5.1.3 HS_CreateHistoryPrivilegeCheck procedure

Purpose

Check whether or not the definer has required privileges on columns composing the primary key of the tracked table and tracked columns.

Definition

```
CREATE PROCEDURE HS_CreateHistoryErrorCheck
   (IN TableName CHARACTER VARYING(<TableNameLength>),
    IN TrackedColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)
   LANGUAGE SQL
   DETERMINISTIC
   READS SQL DATA
   SQL SECURITY INVOKER
   BEGIN
      DECLARE M_TrackedColumns CHARACTER VARYING(<ColumnNameLength>) MULTISET;

      DECLARE i INTEGER;

      --
      -- Construct a multiset consisting of names of tracked columns
      --
      SET i = 1;
      WHILE i <= CARDINALITY(TrackedColumns) DO
         SET M_TrackedColumns[i] =
         HS_ExtractColumnIdentifier(TrackedColumn[i]);
         SET i = i + 1;
      END WHILE;
      --
      -- Check the condition that the multiset of names of
      --   columns composing tracked columns
      --   is contained that of names of columns
      --   on which the definer of a history has SELECT privilege
      --
      IF M_TrackedColumns NOT SUBMULTISET OF
            MULTISET
            (SELECT COLUMN_NAME
               FROM INFORMATION_SCHEMA.COLUMN_PRIVILEGES
               WHERE TABLE_CATALOG = HS_ExtractCatalogIdentifier(TableName)
                  AND TABLE_SCHEMA = HS_ExtractSchemaIdentifier(TableName)
                  AND TABLE_NAME = HS_ExtractTableIdentifier(TableName))
            THEN
         -- Raise an exception condition
         SIGNAL SQLSTATE '2FF26' SET MESSAGE_TEXT =
            'lack of privileges';
      END IF;
   END
```

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

Description

1) The *HS_CreateHistoryPrivilegeCheck(CHARACTER VARYING(<TableNameLength>), CHARACTER VARYING(<ColumnNameLength>) ARRAY)* procedure takes the following input parameters:

   a) a CHARACTER VARYING(<*TableNameLength*>) value *TableName*,

   b) a CHARACTER VARYING(<*ColumnNameLength*>) ARRAY value *TrackedColumns*.

5.1.4 HS_CreateHistoryTableSequenceNumberGenerator procedure

Purpose

Create a sequence generator for generating a sequence number for a history row of the history table.

Definition

```
CREATE PROCEDURE HS_CreateHistoryTableSequenceNumberGenerator
   (IN TableName CHARACTER VARYING(<TableNameLength>))
   LANGUAGE SQL
   DETERMINISTIC
   MODIFIES SQL DATA
   BEGIN
      DECLARE stmt CLOB;
      DECLARE tmpSequenceGeneratorIdentifier
         CHARACTER VARYING(<IdentifierLength>);

      SET tmpSequenceGeneratorIdentifier =
         HS_ConstructSequenceGeneratorIdentifier(TableName);

      SET stmt =
         'CREATE SEQUENCE ' || tmpSequenceGeneratorIdentifier              ||
         ' AS <TypeOf_HS_SEQ> START WITH 1'                                ||
         ' INCREMENT BY 1'                                                 ||
         ' NO MAXVALUE'                                                    ||
         ' NO CYCLE'                                                       ;
      EXECUTE IMMEDIATE stmt;
   END
```

Definitional Rules

1) <TypeOf_HS_SEQ> is the implementation-defined <data type or domain name> of the value generated by the sequence generator.

Description

1) The *HS_CreateHistoryTableSequenceNumberGenerator(CHARACTER VARYING(<TableNameLength>))* procedure takes the following input parameter:

   a) a CHARACTER VARYING(<*TableNameLength*>) value *TableName*.

© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

5.1.5 HS_CreateHistoryTableType Procedure

Purpose

Create the <TableTypeIdentifier> type, which is the base type of a history table corresponding to the specified tracked table.

Definition

```
CREATE PROCEDURE HS_CreateHistoryTableType
   (IN TableName CHARACTER VARYING(<TableNameLength>),
    IN TrackedColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)
   LANGUAGE SQL
   NOT DETERMINISTIC
   MODIFIES SQL DATA
   BEGIN
      DECLARE stmt CLOB;
      DECLARE tmpAllTrackedColumnsDef CLOB;

      CALL HS_CreateCommaSeparatedTrackedColumnAndTypeList(
         TableName, TrackedColumns, tmpAllTrackedColumnsDef);

      -- Construct a CREATE TYPE statement
      --   in order to create <TableTypeIdentifier> type
      --   for a basis of the history table of the specified tracked table.
      SET stmt =
         'CREATE TYPE ' || HS_ConstructTableTypeIdentifier(TableName)       ||
         ' AS(HS_SEQ <TypeOf_HS_SEQ>'                                       ||
         ', ' || tmpAllTrackedColumnsDef                                    ||
         '      , HS_Hist HS_History)'                                      ||
         ' STATIC METHOD HS_HistoryTable()'                                 ||
         '  RETURNS TABLE(HS_SEQ <TypeOf_HS_SEQ>'                           ||
         ', ' || tmpAllTrackedColumnsDef                                    ||
         '      , HS_Hist HS_History),'                                     ||
         ' STATIC METHOD HS_PNormalize('                                    ||
         '     targetColumn CHARACTER VARYING(<ColumnNameLength>))'         ||
         '  RETURNS TABLE(HS_SEQ <TypeOf_HS_SEQ>'                           ||
         ', ' || tmpAllTrackedColumnsDef                                    ||
         '      , HS_Hist HS_History),'                                     ||
         ' STATIC METHOD HS_PNormalize('                                    ||
         '     targetColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)'  ||
         '  RETURNS TABLE(HS_SEQ <TypeOf_HS_SEQ>'                           ||
         ', ' || tmpAllTrackedColumnsDef                                    ||
         '      , HS_Hist HS_History)'                                      ;
      EXECUTE IMMEDIATE stmt;
   END
```

Description

1) The definition of *<TableTypeIdentifier>* type, which is defined in the *HS_CreateHistoryTableType* procedure, is specified in subclause 6.2.1, "<TableTypeIdentifier> Type".

2) The *HS_CreateHistoryTableType(CHARACTER VARYING(<TableNameLength>), CHARACTER VARYING(<ColumnNameLength>) ARRAY)* procedure takes the following input parameters:

   a) a CHARACTER VARYING(<*TableNameLength*>) value *TableName*,

   b) a CHARACTER VARYING(<*ColumnNameLength*>) ARRAY value *TrackedColumns*.

© ISO/IEC 2013 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013

5.1.6 HS_CreateHistoryTable Procedure

Purpose

Create a history table corresponding to the specified tracked table.

Definition

```
CREATE PROCEDURE HS_CreateHistoryTable
   (IN TableName CHARACTER VARYING(<TableNameLength>),
    IN TrackedColumns CHARACTER VARYING(<ColumnNameLength>) ARRAY)
   LANGUAGE SQL
   NOT DETERMINISTIC
   MODIFIES SQL DATA
   BEGIN
      DECLARE stmt CLOB;
      DECLARE tmpAllIdentifierColumns CLOB;
      DECLARE tmpAllIdentifierColumnsOpt CLOB;

      CALL HS_CreateCommaSeparatedIdentifierColumnList(
         TableName, TrackedColumns, NULL, tmpAllIdentifierColumns);
      CALL HS_CreateCommaSeparatedIdentifierColumnList(
         TableName, TrackedColumns, ' WITH OPTIONS NOT NULL',
         tmpAllIdentifierColumnsOpt);

      -- Construct a CREATE TABLE statement
      --   in order to create the history table
      --   according to the specified tracked table.
      SET stmt =
         'CREATE TABLE ' || HS_ConstructTableIdentifier(TableName)          ||
         '  OF ' || HS_ConstructTableTypeIdentifier(TableName)              ||
         '  (REF IS "HS_REF" SYSTEM GENERATED,'                             ||
               tmpAllIdentifierColumnsOpt || ','                            ||
         '    PRIMARY KEY(HS_SEQ)'                                          ||
         '  )'                                                              ;
      EXECUTE IMMEDIATE stmt;
   END
```

Description

1) The *HS_CreateHistoryTable(CHARACTER VARYING(<TableNameLength>), CHARACTER VARYING(<ColumnNameLength>) ARRAY)* procedure takes the following input parameters:

   a) a CHARACTER VARYING(<*TableNameLength*>) value *TableName*,

   b) a CHARACTER VARYING(<*ColumnNameLength*>) ARRAY value *TrackedColumns*.

5.1.7 HS_CreateInsertTrigger Procedure

Purpose

Create a trigger to insert a history row into the history table when a row is inserted into the corresponding tracked table.

Definition

```
CREATE PROCEDURE HS_CreateInsertTrigger
```


© ISO/IEC 2013 – All rights reserved
Documents
Document Sharing Hub
Full document details: PD ISO/IEC TS 13249-7:2013