Function comparison among SQL/MM History, SQL/

	Function		SQL/MM History	SQL/Temporal	
				ISO	ANSI
1	Data type for period		User-defined structured type HS_History	Constructed type PERIOD	
2	SURROGATE type□□□□□□□□□□□□□□(Data type for unique values generated by system)		Possible to describe equivalently using sequence	Outside the scope of SQL/Temporal	
3	Time granularity	Datetime types	YEAR, MONTH, DAY, HOUR, MINUTE, SECOND	Can be handled for all of datetime types	
		Other than built-in datetime types	FISCAL_YEAR, HALF_YEAR, QUARTER, WEEK	Not supported	
4	Time granularity in interval qualifier		Not supported by SQL	Not supported	
5	Timestamp qualifier		Not supported by SQL	Not supported	
6	Indeterminate datetime type		Not supported by SQL	Not supported	
		Timestamp type	Timestamp type with scale zero and without time		

7	Element type of period type	Other than timestamp type	Possible to describe equivalently with SQL for datetime type provided by SQL *1	All of datetime types can be element type
		With time zone	Possible to describe equivalently with SQL *2	
		Exact numeric type	Possible to describe equivalently with SQL *3	Exact numeric type with scale zero
8	Implemantation-defined calender		Not supported	Not supported
9	Calender-property spec	Calender spec clause	Not supported	Not supported
		Property spec clause		Not supported
		Property table		Not supported

10	State table		Can be handled by history table	Not supported
11	Event table		Not supported	Not supported
12	Table with transaction time	With valid time	Can be handled by history table	Not supported
		Without valid t	Not supported	
13	Addition and substraction of period type and interval type		Possible to describe equivalently with	period-value + interval-value
				period-value - interval-value
14	Symbolic time zone		Not supported by SQL	Not supported
15	WEIGHTED option in set function		Possible to describe equivalently with SQL *5(a lot of codes)	Not supported

16	Set function RISING		Possible to describe equivalently with SQL *6(a lot of codes)	Not supported	
17	Proximity function	Next value of datetime vakue by one granule	Possible to describe equivalently with SQL *7	NEXT(datetime-value)	
		Prior value of datetime vakue by one granule		PRIOR(datetime-value)	
18	Period bound function	First value(beginning value)	period-value. HS_HistoryBeginTime	FIRST(period-value)	BEGIN(period-value)
		Last value	period-value. HS_HistoryEndTime	LAST(period-value)	
		Next value of the last value	Possible to describe equivalently with SQL *8	Not supported	END(period-value)
		Ending value	Next value of the last value	Maximum value of datetime in	Next value of the last value
19	Operation to instant set	First value	Not necessary	Not necessary	
		Last value			
		Union operation			
		Except operation			
		Intersect operation			

20	Operation to temporal	intersect operation	Not necessary	Not necessary
21	Interval value for length of period	Month interval	period-value. HS_MonthInterval(	INTERVAL(period-value) Month
		Day interval	period-value. HS_DayInterval□	INTERVAL(period-value) Day
		Other interval	Possible to describe equivalently with SQL *9(a lot of codes for general definition)	INTERVAL(period-value) interval qualifier
22	Period constructor	Specify beginnig datetime and last datetime	HS_History(beginning-timestamp-value,ending-timestamp-value)	PERIOD[beginning-datetime-value, ending-datetime-value)
		Specify upper or lower bound	Possible to describe equivalently with SQL *10	PERIOD(beginning-datetime-value, ending-datetime-value]
				PERIOD(beginning-datetime-value, ending-datetime-value)
				PERIOD[beginning-datetime-value, ending-datetime-value]
23	Period literal		Use period constructer	PERIOD period-string (in addition to the format of period constructor minus sign(-) can be specified as a delimiter between beginning datetime and
	Datetime literal /	Calender - property spec	Not supported by	
		Format of calender		

24	Interval literal	Indeterminate datetime / interval	Not supported by SQL	Not supported by SQL
		Now-relative		
25	Special value of datetime			
		Initiation	Not supported	Not supported
		Until changed	Represented by	Not supported
		Beginning	Use minimum representable datetime value	Use minimum representable datetime value
		Forever(maximum datetime)	Use maximum representable datetime value	Use maximum representable datetime value
		All of time	Not supported	Not supported
26	Operation betwween period values	Union	period-value.HS_Union(period-value)	period-value P_UNION period-value
		Except	period-value.HS_Except(period-value)	period-value P_EXCEPT period-value
		Intersect	period-value.HS_Intersect(period-value)	period-value P_INTERSECT period-value
		Between period type and character string type	Possible to describe equivalently with SQL *11	Data conversion for character string whose format is as 'begin datetime,end datetime'

27	CAST specification	Between period type and date type	Possible to describe equivalently with SQL *12	Data conversion date value to period value that has one element whose value is the casted date value
		Between period type and timestamp type	Possible to describe equivalently with SQL *13	Data conversion timestamp value to period value that has one element whose value is the converted timestamp value
		Between period types	Not necessary*14	Cast for begin datetime value and end datetime value
		Between period type and numeric type	Not supported	Not supported
		Between period type and temporal	Not necessary	Not necessary
		Between temporal element and	Not necessary	Not necessary
		Time granularity as CAST target	Granularity is not provided as data type	Not supported
28	SCALE operation ☐change of time granularity☐		Granularity is not provided as data type	Not supported

29	NOBIND function		Not supported	Not supported	
30	NORMALIZE function (devide set of period values to connected components with regard to whether or not periods overlap, and then get a union of period values for		Possible to describe equivalently with SQL *15(a lot of codes)	Not supported	NORMALIZE(period-set-value)
31	EXPAND function (get a set of period values such as each period has one element datetime value		Possible to describe equivalently with SQL *16(a lot of	Not supported	EXPAND(period-value)
32	Comparison predicate			Comparison of row values whose row value elements are FIRST(period value) and LAST(period	Comparison of row values whose row value elements are BEGIN(period value) and END(period value)
		Equality comparison	period-value. HS_Equal(begin timestamp value,end timestamp value)	period-value comparison-operator period-value	
		Other conparisons	Possible to describe equivalently with SQL *17		
		OVERLAPS	period-value. HS_Overlaps(begin timestamp value,end timestamp value)	period-value INTERSECTS period value	period-value OVERLAPS period-value

33	Period comparison	PRECEDS	period-value. HS_Precedes(begin timestamp value,end timestamp value)	period-value PRECEDES period-value
		SUCCEEDS	period-value. HS_Succees(begin timestamp value,end timestamp value)	period-value SUCCEEDS period-value
		MEETS	period-value. HS_Meets(begin timestamp value,end timestamp value)□□□□□□□□□□□□□□[test if end timestamp value and begin	period-value MEETS period-value□□□[test if NEXT(end timestamp value) and begin timestamp value are equal]
		CONTAINS	period-value. HS_Contains(begin timestamp value,end timestamp value)	period-value CONTAINS period-value
34	PLAUSIBILITY		Not supported	Not supported

35	Snapshot		Not necessary	Not necessary
36	Selection of the row whose datetime or period is specified value		Possible to describe equivalently with SQL *18	Use serch condition
37	Normalized query specification		Invoke table function HS_Period(ARRAY['history column',...]) as a table reference in FROM clause	query-specification NORMALIZE ON column-name[, column-name]…□□□□□□□□□□□□□□□□□□□□ □normalize set of rows whose values of non-normalizing column are equal each other with respect
		Normalizing column	One column of HS.History	More than one normalizing column are allowd
		History column(value-equivalent column)	History column	Other than normalizing columns
38	Shorthand syntax for multiple corelation name to one table		Use normal SQL syntax	Outside the scope of SQL/Temporal
39	Reference to valid time		Attribute reference to	Valid time is not provided
40	Reference to transaction time		Attribute reference to	Transaction time is not provided

41	EXPANDING clause on query expression (merge sets of rows with period)		Possible to describe equivalently with SQL *20(a lot of codes)	query-expression {ALL | EXPANDING(column-name[, column-name]...□query-expression□□expand period value of every query-expression, and
42	Grouping based on time granule			Not supported
		Granule used for grouping	Possible to describe equivalently with SQL for some case *21(a lot of codes)	
		Start aggregate		
		End aggregate		
43	Getting valid time		Attribute reference to	
44	Setting valid time on insert		Use static method HS_HistoryBeginTime and HS_HistoryEndTime	
45	Setting valid time on update		Use static method HS_HistoryBeginTime and HS_HistoryEndTime	
46	Deletion of the row of specified valid time		Not allowed	

47	Invalidation rows for removal of expired rows[VACUUM]		Not supported	Not supported
48	Genaration of value of SURROGATE type		Use NEXTvalue expression for sequence	Outside the scope of SQL/Temporal
49	Non-applicable column		Use generated column	Outside the scope of SQL/Temporal
50	Default value for datetime or interval		Values provided by SQL	Values provided by SQL
51	Distribution □over indeterminate type value□	Definition	Indeterminate type is not supported	Indeterminate type is not supported
		Drop		
		Alter		
52	Alter table	Add valid time	by definition of history	Not supported
		Drop valid time	by dropping	
		Alter valid time	Not supported	
		Add transaction time	Not supported	
		Drop transaction	Not supported	
		Definition of SCALE	Not supported	

		Definition of CAST	Not supported		
		Addition and alternation VACUUM	Not supported		
53	Calender declaration		Not supported	Not supported	
54	Setting of session information	Property setting	Not supported	Not supported	
		Schema version			
		CREDIBILITY			
		PLAUSIBILITY			
		SCALE			
		CAST			
55	Descriptor of dynamic SQL and CLI		Provoded as information of userdefied type by information	Item information of SQL descriptor and CLI descriptor	

Temporal and TSQL2

TSQL2
Constructed type PERIOD
Data type SURROGATE
Can be handled for all of datetime types
Implementaion-defined datetime types
Time granularity can be specified as datetime field(Time granularity is identified by time
Same as interval identifier
[NOSTANDARD] [GENERAL] INDETERMINATE datetime-type

All of datetime types with or without timestamp qualifier can be element type
Not supported
Implemantation-defined calender can be referenced by calender-property spec
WITH CALENDRIC calender-spec (the data conversion between datetime value and calender value)
WITH PROPERTIES property-spec□□□(the format of calender value)
Table that has information about formats for calender

Add the following clause to table definition: AS VALID STATE [timestamp identifier] [AND TRANSACTION] □□□□□ □□□□□□□□□ (a collection of periods, called temporalelement is
Add the following clause to table definition: AS VALID EVENT [timestamp identifier] [AND TRANSACTION] □□□□□ □□□□□□□□□ (a collection of timestamps, called instant set is added
Add the following clause to table definition: AS TRANSACTION (the timestamp when inserted is added to each row)
period-value + interval-value
eriod-value - interval-value
'MST' etc
set-function-type([set-quantifier] [WEIGHTED] argument) (aggregate by multiplying the number of granules contained in temporal element to the

RISING([set-quantifier] [WEIGHTED] argument) □□□□□□□□□□□□□□□□□□□□□□□□ (calculate the longest period such as
Not supported
BEGIN(period-value)
END(period-value)
FIRST□instant set value□
LAST□instant set value□
instant set value + instant set value
instant set value - instant set value
INTERSECT(instant set value, instant set value)

INTERSECT(temporal element value, temporal element
INTERVAL(period-value)
PERIOD(beginning-datetime-value, ending-datetime-value)
Not supported
PERIOD period string (period string has a format as registered in a property table)
Calender - property spec
Format as registered in property table

Representation such as 'aaaa-bb-cc□dddd-ee-ff'
now_string property(representation such
Values registered in property table represents special vale
initiation_string property
until_changed_string property
beginning_string property
forever_string property
'All of time'
Not supported
Not supported
INTERSECT(period-value, period-value)
Data conversion based on format of calender

Data conversion date value to period value that has one element whose value is the casted date value
Data conversion timestamp value to period value that has one element whose value is the converted timestamp
Cast for begin datetime value and end datetime value□the result of cast from coarser granule to finer granule is the first granule in finer
Data conversion based on format of calender
Cast to temporal element after cast between period values
Extract the first instant of temporal element
CAST(datetime-value AS time-granularity-identifier)
SCALE(source-value AS [datetime-type | time-granularity-identifier]) □□□□□□□ □□□□the result of cast from coarser

NOBIND(datetime-value)
Not supported
Not supported
□
period-value comparison-operator period-value
row-value OVERLAPS row-value

row-value PRECEDES row-value
row-value SUCCEEDS row-value
row-value MEETS row-value
row-value CONTAINS row-value
Add the following clause to WHERE clause:□□□□□ □□□□□□WITH PLAUSIBILITY integer value (within range from 0 to 100)

SELECT[set-quantifier] SNAPSHOT select-list…
Specify the following VALID clause immediately before FROM clause:□□□□□□□□□□□□□□□□{VALID | VALID INTERSECT}{temporal-element-
Specify the following clause as table reference:□□□□□□□□□{table-name | corelation-name}[{ (coalescing-column[, coalescing-column]… | * })][({PERIOD |
Equvalent to one normalizing column
Coalescing column
{table-name | corelation-name} [AS] corelation-name[corelation-name]…
VALID([table-name | corelation-name])
TRANSACTION([table-name | corelation-name])

Not supported
GROUP BY {VALID(table-name or corelation-name) | column reference } [USING clause][LEADING clause][TRAILING clause]
USING {integer | time-granule |integer time-granule | PERIOD 'All of time'}
LEADING {integer | time-granule |integer time-granule | PERIOD 'All of time'}
TRAILING {integer | time-granule |integer time-granule | PERIOD 'All of time'}
FETCH…INTO□ INTO VALID[PERIOD] target list
Specify VALID clause on INSERT statement
Specify VALID clause on UPDATE statement
Specify VALID clause to DELETE statement

Add the following clause to table definition:□□□□□□□□
NEW specification as insert column
Column definition as follows:□□□□□column-name data-type INAPPLICABLE value-expression
Value with a calender format also can be defualt value
CREATE DISTRIBUTION [{GLOBAL | LOCAL} TEMPORARY] DISTRIBUTION distribution-name USING table-name
DROP DISTRIBUTION DISTRIBUTION distribution-
ALTER DISTRIBUTION DISTRIBUTION distribution-
ADD [VALID] {STATE | EVENT} [timestamp-qualifier]
DROP [VALID] {STATE | EVENT}
REPLACE [VALID] {STATE | EVENT} [timestamp-qualifier]
ADD TRANSACTION
DROP TRANSACTION
SCALE VALID AS timestamp-qualifier

CAST VALID AS timestamp-qualifier
☐VACUUM datetime-value
DECLARE CALENDAR SYSTEM WITH calender-spec
SET PROPERTIES[FOR CHARACTER SET …] [FOR { granularity identifier | calender name } WITH property spec
SET SCHEMA datetime-value
SET CREDIBILITY {integer value | AS DEFAULT}
SET PLAUSIBILITY {integer value | AS DEFAULT}
SET SCALE {granularity identifier | AS DEFAULT}
SET CAST {granularity identifier | AS DEFAULT}
?