Standardkonforme Integration quelloffener Big Data-Lösungen
in existierende Netzleitsysteme

Anfragesprache für Zeitreihen
Time Series Query Language
Version 1.0.0

Editor

Andre Göring (OFFIS)

Contributors

Michael Brand (OFFIS), Jad Asswad (OFFIS), Patrick Bruns (OFFIS), Norman Ihle (OFFIS), Michael Specht (KISTERS), Carsten Saathoff (KISTERS), Bernd Grünefeld (KISTERS), Jörg Friebe (KISTERS), Tim Rasim (IMIS), Marcel Winter (IMIS)

OFFIS: OFFIS – Institut für Informatik

KISTERS: KISTERS AG

IMIS: Institut für Multimediale und Interaktive Systeme – Universität zu Lübeck

10th February, 2020
Revision 1.0.0

Document History

Version	Date	Author(s)	Description/Comments
0.5	2018-09-29	Andre Göring	Creation of Document and first version according to NetzDatenStrom interfaces discussions.
0.5.1	2019-04-02	Andre Göring	Fixing parser-discussion changes: adding “” for Aliases, adding Brackets () for Interpolate/Aggregate/Raster parameters. Adding “Type” description.
1.0	2020-10-02	Andre Göring	Final Editing for Publication

Abstract

Im Projekt NetzDatenStrom werden zwei unterschiedliche Archive zur Haltung von Zeitreihen verwendet. Um diese untereinander austauschen zu können, wurde festgestellt, dass eine einheitliche Anfragesprache für Zeitreihen sinnvoll ist. Auf der einen Seite soll dadurch eine saubere Trennung zwischen Modulen, welche Zeitreihenwissen benötigen, und zwischen Archiven, welche Zeitreihenwissen zur Verfügung stellen, sichergestellt werden. Auf der anderen Seite soll vermieden werden, für Module eine oder mehrere modulspezifische proprietäre Schnittstelle(n) im Archiv umzusetzen. Die Entwicklung der Archive soll somit von der Entwicklung der Module entkoppelt werden, und der Zugriff von Modulen nur über die Anfragesprache erfolgen. Dieses Dokument beschreibt Anforderungen, Syntax, Semantik, Rückgabe und Beispiele für die Zeitreihenanfragesprache, welche im Projekt NDS in Teilen umgesetzt, aber für die allgemeine Verwendung definiert wird.

Inhalt

1 Anforderungen an die Zeitreihenanfragesprache

Im Projekt NetzDatenStrom (NDS) wird die Ausgabe von Zeitreihendaten aus einem austauschbaren Big-Data-Archiv für Zeitreihen benötigt. Die Verwendung einer einheitlichen Anfragesprache soll allerdings nicht auf NDS eingeschränkt werden, sondern auch darüber hinaus sinnvoll möglich sein. Deshalb ergeben sich aus den Überlegungen zu NDS, openKONSEQUENZ (openKONSEQUENZ, 2018) und der Erfahrung von NDS-Projektteilnehmern im Bereich Big-Data-Archivierung folgende Anforderungen für die Anfragesprache die nachfolgend beschrieben werden.

- Einfache Verwendbarkeit für den Anwender
- Zeitreihenspezifika herausstellen
- Wiederverwendbarkeit für verschiedene Datenbanktypen für Zeitreihen
- Offen für Erweiterungen
- Lesender, nicht ändernder Zugriff (die Befüllung der Datenbank ist nicht Bestandteil dieser Dokumentation – gegebenenfalls ist es sinnvoll, die Anfragesprache dementsprechend später zu erweitern)

1.1 *Einfache Verwendbarkeit für den Anwender*

Die Anfragesprache für Zeitreihen soll über NDS hinaus verwendet werden können. Als wichtiges Merkmal für die Wiederverwendung ist die (Wieder-)Verwendbarkeit. Insbesondere sollen Entwickler von Modulen oder Anwender möglichst einfach, ohne eine lange Einarbeitungszeit Anfragen an das/die Big-Data-Archive stellen können. Insbesondere soll die Anfragesprache menschenlesbar, leicht verständlich und soweit möglich ohne kompliziert verschachtelte Struktur verwendbar sein.

1.2 *Zeitreihenspezifika herausstellen*

Die Anfragesprache wird auf Zeitreihen eingeschränkt. Die wichtigen Merkmale von Zeitreihen sind die Zeitintervalle oder Zeitpunkte von in Zeitreihen gespeicherten Werte. Diese sollen möglichst einfach für den Benutzer angegeben werden können. Darüber hinaus soll es ermöglicht werden, anhand der Stützwerte der Zeitreihen durch Interpolation, Aggregation oder neuer Rasterung auch Zwischen/Gesamtwerte berechnen zu lassen. Letztere Operationen sind aus mehreren Gründen sinnvoll:

- Ermöglichung eines Vergleich von Messreihen mit unterschiedlicher Auflösung und Raster
- Ermöglichung sinnvoller mathematischer Operationen auf Messreihen mit unterschiedlicher Auflösung und Raster
- Beschränkung der Ausgabe auf relevante bzw. benötigte Werte

Einige dieser Funktionen finden sich beispielsweise auch in openTSDB – einer open source Datenbanktechnologie für Zeitreihen (The OpenTSDB Authors, 2018) – welche jedoch eine proprietäre Anfragesprache für Zeitreihen umsetzt.

1.3 *Wiederverwendbarkeit für verschiedene Datenbanktypen für Zeitreihen*

Durch die Anfragesprache soll die Art der Datenbank nicht eingeschränkt werden. Jede beliebige Datenbank soll für die Speicherung und Lieferung von Zeitreihen verwendet werden können. Im Kontext von Big-Data soll die langfristige Speicherung und Verwendbarkeit der Daten im Vordergrund stehen. Hier können unterschiedliche Anforderungen bei der Wahl der Technologie zu unterschiedlichen Archivarchitekturen führen. Die Anfragesprache soll weitestgehend neutral zu dieser Technologieauswahl stehen und diese möglichst wenig beeinflussen. Außerdem sollen auch schon existierende Zeitreihen-Archive, welchen unterschiedliche Technologien zugrunde gelegt sind, die Anfragesprache umsetzen können, um eine Austauschbarkeit sicherzustellen.

1.4 *Offen für Erweiterungen*

Im Projekt NDS wurde eine Vielzahl an fachlichen Aufgaben einer Anfragesprache diskutiert, priorisiert und aus der Sicht der Mitarbeiter an NDS in eine sinnvolle Umsetzungsreihenfolge gebracht („Zeitreihenanfragesprachen-Level“). Die Liste der Aufgaben wird im Dokument „NetzDatenStrom AP2 Dokumentation“ beschrieben und gibt eine Möglichkeit der Implementierungsreihenfolge vor. Die im vorliegenden Dokument beschriebene Anfragesprache wird jedoch unabhängig von einer Implementierung(sreihenfolge) definiert, damit eine solche die Ausprägungen und damit die Verwendbarkeit der Anfragesprache nicht negativ einschränkt beziehungsweise nicht schrittweise vorzeichnet. Die Vollständigkeit einer solchen Liste für Zeitreihenanfragesprachen „sinnvoller“ Aufgaben, kann und möchten die Mitarbeiter im Projekt NDS jedoch nicht garantieren. Deshalb soll die Anfragesprache so angelegt werden, dass eine Erweiterung möglich ist. Dies spiegelt sich auch in der folgenden Anforderung wieder.

1.5 *Lesender, nicht ändernder Zugriff*

Die hier dokumentierte Anfragesprache ist zum jetzigen Zeitpunkt auf einen lesenden Zugriff auf Zeitreihenarchivdaten beschränkt. Die Befüllung der Datenbanken ist nicht Bestandteil dieser Dokumentation – gegebenenfalls ist es sinnvoll, die Anfragesprache dementsprechend später zu erweitern.

2 Lösung

Die neu definierte Zeitreihenanfragesprache basiert auf SQL (ISO/IEC, 2016). SQL ist vielen Entwicklern bereits bekannt und ist menschenverständlich lesbar. Damit ist eine einfache Verwendbarkeit durch den Anwender gegeben. Für Zeitreihenspezifika werden folgende Erweiterung/Veränderung vorgesehen:

- FROM Table wird durch FROM Type ersetzt. Das Table impliziert die Verwendung einer relationalen Datenbank. Die Anfragesprache soll jedoch unabhängig vom Datenbanktyp agieren. Eine Einschränkung des Types soll dennoch möglich sein, um eine Ausgabe in bestimmter Form erwarten zu können und festzulegen welche Werte in dem Typen jeweils eigene Zeitreihen abbilden.
- Die Zeiteinschränkung des auszulesenden / auszugebenden Zeitintervalls bzw. Zeitwerts erfolgt durch eine besondere Notation direkt hinter dem Typen.
- Zeitintervalle können neu abgetastet werden. Dazu werden 3 besondere Maßnahmen benötigt:
    - Interpolation für eine höhere Abtastrate als ursprünglich abgelegte Messwerte,
    - Aggregation für eine niedrigere Abtastrate als ursprünglich abgelegte Messwerte,
    - Raster für eine mögliche zeitliche Verschiebung des Rasters der ursprünglich abgelegten Messwerte.
- Einschränkung der Ausgabe auf eine festgelegte Anzahl an Werten

Die Erfüllung der Anforderungen wird im Folgenden diskutiert

2.1 *Einfache Verwendbarkeit für den Anwender*

Durch die Verwendung einer an SQL angelegten Syntax und Semantik ist ein breites Vorwissen im IT-Bereich zu erwarten. Des Weiteren ist SQL menschenlesbar, verständlich und weitestgehend ohne verschachtelte Klammerstruktur verwendbar.

2.2 *Zeitreihenspezifika herausstellen*

Durch das Hinzufügen der Auswahl von Zeitpunkten / Zeitintervallen, sowie verschiedenen Möglichkeiten zur Zusammenfassung und Neuberechnung von Stützstellen, rückt die Anfragesprache Zeitreihen in den Mittelpunkt. Benötigte Werte sind so vereinfacht abrufbar.

2.3 *Wiederverwendbarkeit für verschiedene Datenbanktypen für Zeitreihen*

Die Überdeckung des SQL „FROM Table“ durch „FROM Type“ suggeriert bereits eine Unabhängigkeit von relationalen Datenbanken. Der Typ beschreibt die Art der Daten, die in der Datenbank vorliegen und abgefragt werden soll. Dies dient der Einschränkung der Rückgabe, so dass die Datenabfrage Informationen in einem zu erwartenden Format zurückgeben kann. Hinsichtlich der an SQL angelehnten Anfragesprache ergibt es sich jedoch, dass die Anbindung von Archiven an die Anfragesprache je nach zugrundeliegender Datenbanktechnologie verschieden kompliziert ausfallen kann. Dies lässt sich aufgrund der Vielfachheit an existierenden Anfragesprachen und Anfragearten nicht verhindern. Sinnvoll ist es, je angebundener, zugrundeliegeder Datenbanktechnologie einen Adapter zu erzeugen, der die Ein- und Ausgabe der Anfragesprache entsprechend dieser Dokumentation umsetzt. Falls dies möglich ist, kann ein ansonsten tieferer Eingriff in die zugrundeliegende Datenbank vermieden werden.

2.4 *Offen für Erweiterungen*

Die an SQL angelehnte Syntax ermöglicht die Erweiterung der Anfragesprache durch weitere Schlüsselwörter für gänzlich neue Funktionalitäten. Dies wird bei SQL auch hinsichtlich der Einschränkung der Anzahl der Rückgabewerte beispielsweise mittels „LIMIT“ oder „TOP“ für verschiedene SQL-Bibliotheken unterschiedlich gehandhabt. Hinsichtlich der Interpolation, Aggregation, Rasterung von Daten, aber auch der mathematischen sowie logischen Infix und Präfixfunktionen ist die Anfragesprache offen definiert. Sodass jederzeit eine Erweiterung solcher Funktionen erfolgen kann. Damit die entsprechende Funktionalität zur Verfügung steht, muss für die zugrundeliegende Technologie dann eine spezifische Umsetzung erfolgen.

2.5 *Lesender, nicht ändernder Zugriff*

Der Fokus der Umsetzung liegt auf der Abfrage von in Archiven befindlichen Daten. Diese Daten werden den Archiven derzeit nicht über eine einheitliche Anfragesprache sondern über weitere Schnittstellen zur Verfügung gestellt. Deshalb wird insbesondere das SQL Statement „SELECT“ hier verwendet, abgeändert und erweitert. Insbesondere langfristig ist das Erzeugen, Überschreiben oder Entfernen von Daten über eine einheitliche Anfragesprache für Zeitreihen sicherlich interessant, steht aber nicht im Fokus dieser Arbeit, sodass eine Diskussion der SQL-Schlüsselworte „DROP“, „UPDATE“, „DELETE“ sowie „INSERT“ hier zum jetzigen Zeitpunkt ausbleiben muss.

3 Syntax & Semantik

Wie im vorherigen Kapitel beschrieben, beruhen Syntax und Semantik der Anfragesprache auf denen von SQL.

Im Folgenden werden die Änderungen gegenüber SQL und die wesentlichen neuen Funktionalitäten beschrieben. Die Syntax wird in Syntaxdiagrammen dargestellt und die Semantik im Anschluss diskutiert. Abgerundete Kästen stellen in der Syntax-Visualisierung Schlüsselwörter dar (Bezeichner im Text in „“), die Definition eckiger Kästen (Bezeichner im Text *kursiv*) werden nachfolgend besprochen. Wesentliche neue Anfragesprachen-Elemente sind orange eingefärbt. Eine Darstellung der Syntax in erweiterter Backus Naur Form (eBNF) (ISO/IEC, 1996) findet sich im Anhang.

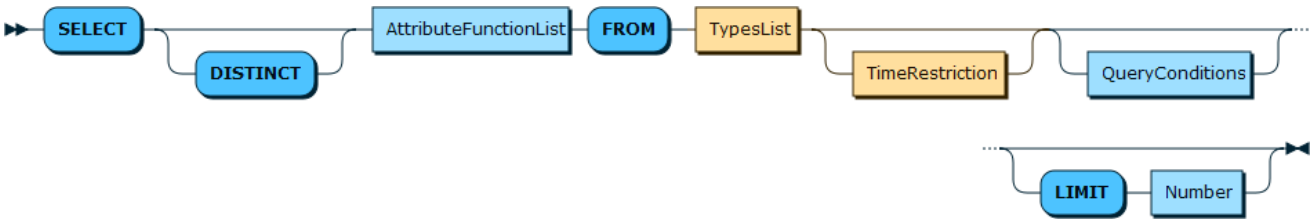
Schlüsselwörter sind an dieser Stelle der einfachen Lesbarkeit halber nur in Großbuchstaben notiert. Jede mögliche Groß- und Kleinschreibung der Schlüsselwörter sollte jedoch akzeptiert und dementsprechend verarbeitet werden.

Nummern, Namen oder feste Werte sind in der eBNF nur als Schlüsselwörter „NUMBER“, „NAME“ bzw. „VALUE“ dargestellt. Entsprechende Umsetzungen sind äquivalent zu SQL zu gestalten. Die Beschreibung der Syntax/Semantik wird in der folgenden Vorstellung an solchen Stellen abgebrochen, wo eine Nummer als *Number,* ein fester Wert als *Value* oder *Name* im Bezeichner der eBNF-Definiton vorkommt.

Beispiele dazu finden sich im Kapitel 5.

3.1 *SelectQuery*

SelectQuery:



Für lesende Anfragen an das Archiv wird ein „SELECT“ vorangestellt und beschrieben, welche Attribute zurückgegeben oder vor der Rückgabe mit mathematischen Funktionen weiterverarbeitet werden sollen (siehe nachfolgende *AttributeFunctionList*). Diese *AttributeFunctionList* hat den gleichen Aufbau wie in SQL. Es ist jedoch das Einfügen von Zeitreihenspezifischen Aggregationen möglich.

Das Schlüsselwort „DISTINCT“ ist gleich wie in SQL – es entscheidet darüber, ob – falls vorhanden – mehrere gleiche Zeitreiheneinträge zurückgegeben werden (durch Weglassen des Schlüsselwortes „DISTINCT“) oder ob gleiche Zeitreiheneinträge in der Auswertung / Ausgabe zu einer zusammengefasst werden sollen (durch Hinzufügen des Schlüsselwortes „DISTINCT“).

Durch das Schlüsselwort „FROM“ wird die Art der Typen gewählt (siehe Nachfolgende *TypesList*), welche ausgewertet werden sollen. Hier wird das SQL-Schlüsselwort „FROM“ überdeckt, welches für SQL Tabellen wählt. Eine genaue Beschreibung, welche Informationen für einen Typen bekannt seien müssen, findet sich in Anhang *8.5.*

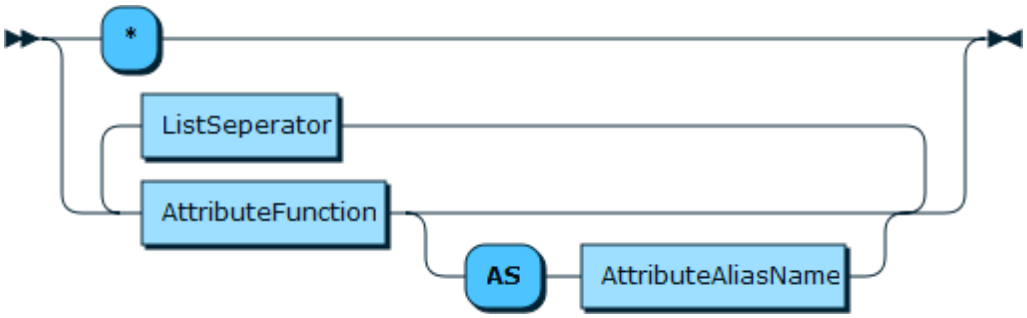
Die optionale *TimeRestriction* (siehe nachfolgend) enthält die Einschränkung für welchen Zeitraum Daten zurückgeliefert werden sollen.

Die ebenfalls optionalen *QueryConditions* (siehe nachfolgend) schränken die Auswahl von Zeitreiheneinträgen auch für über die Zeit hinausgehende Attribute ein und werden von SQL übernommen.

Das optionale Schlüsselwort „LIMIT“ limitiert die Ausgabe auf eine feste Anzahl von Zeitreiheneinträgen entsprechend der *Number.* Dieses Schlüsselwort ist auch in einigen SQL-Dialekten zu finden – jedoch nicht Bestandteil des SQL Standards.

3.2 *AttributeFunctionList*

AttributeFunctionList:

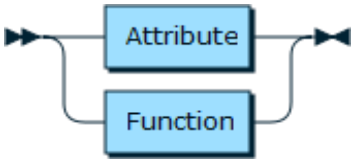


Die Liste der zurückzugebenden ausgewählten Attribute hat verschiedene mögliche Ausprägungen. Das Schlüsselwort „*“ ist wie in SQL belegt mit der Rückgabe aller Attributarten (die dem „FROM“ Typ entsprechen).

Verschiedene Attribute beziehungsweise Funktionen (siehe nachfolgende *AttributeFunction*) können durch den Listen-Seperator (*ListSeperator*, ein „Komma“) ausgewählt und optional mit dem Schlüsselwort „AS“ mit neuem Namen versehen werden. Hierbei können diese neue Namen auch in „doppelten Anführungszeichen“ gesetzt werden, um auch Sonderzeichen verwenden zu können und eine etwaige mathematische Auswertung zu verhindern.

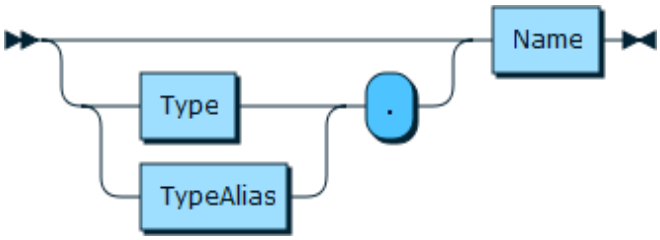
3.3 *AttributeFunction, Attribute und AttributeName*

AttributeFunction:



Attribute:

AttributeName:

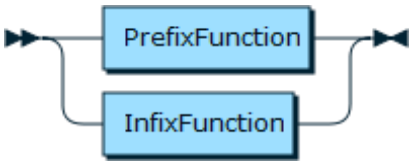


AttributeFunction differenzieren sich in *Attribute* und *Function* – also der Auswahl von einzelnen Attributen (*Attribute*) oder der Auswahl von auf Attributen durchzuführenden mathematischen Funktionen (siehe nachfolgend *Function*). Hierbei setzt sich der Attributname zusammen aus dem Typnamen (siehe nachfolgend *Type*) oder dem Alias eines Typen (siehe nachfolgend *TypeAlias*), und dem Attributnamen eines Attributes des vorgenannten Typs. Beide Namen werden punktsepariert „.“. Wird nur ein *Type* verarbeitet, so kann der *Type* bzw. *TypeAlias* und der „Punkt“ weggelassen werden.

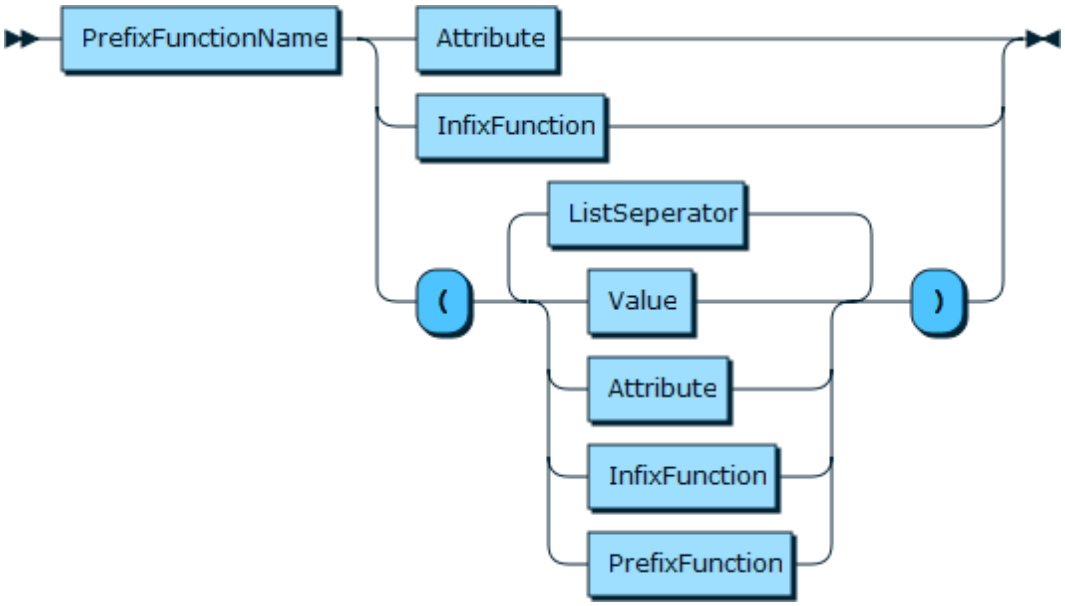
Attributenamen sind über die Archiv-interne Bezeichnung aufzurufen und auszugeben oder mit Alias umzubenennen.

3.4 *Function - PrefixFunction*

Function:



PrefixFunction:



Auf Attribute durchgeführte mathematische Funktionen sind entweder Prefix-Funktionen oder Infix-Funktionen. Prefix-Funktionen haben einen Bezeichner (*PrefixFunctionName*). Prefix-Funktionen haben zwei Funktionsweisen:

1. Sie können Attribute verschiedener Zeitreiheneinträge zu einem Ergebniswert zusammenfassen oder aber weitere Infix-Funktionen (siehe nachfolgend *InfixFunction*) über alle Anfrage-Ergebnisse zu einem Ergebniswert zusammenfassen. Dies entspricht einer Aggregation von Zeitreihenwerten.
2. Sie können auf einzelne Werte (*Value*), Attribute (*Attribute*), Infix-Funktionen (*InfixFunction*) sowie weitere Prefixfunktionen (*PrefixFunction*), optional mehrfach Listenseparіert (*ListSeperator*, ein „Komma“) mit weiteren Parametern aus der genannten Liste angewendet werden. Dies entspricht höheren Mathematischen Funktionen auf einem oder mehreren Werten eines Zeitreiheneintrages.

Mögliche Prefix-Funktionen und ihre *PrefixFunctionName-Schlüsselworte* sind in folgender, erweiterbaren Liste aufgeführt:

Prefix-Funktion	**PrefixFunctionName**	**Beschreibung**
Minimum	MIN	Gibt das Minimum aller über die verschiedenen Objektinstanzen gesammelten Attribute mit gleichem Attributnamen zurück.
Maximum	MAX	Gibt das Minimum aller über die verschiedenen Objektinstanzen gesammelten Attribute mit gleichem Attributnamen zurück.
Durchschnitt	AVG	Gibt den Durchschnitt der über die verschiedenen Objektinstanzen gesammelten Attribute mit gleichem Attributnamen zurück.
Summe	SUM	Gibt die Summe der über die verschiedenen Objektinstanzen gesammelten Attribute mit gleichem Attributnamen zurück.

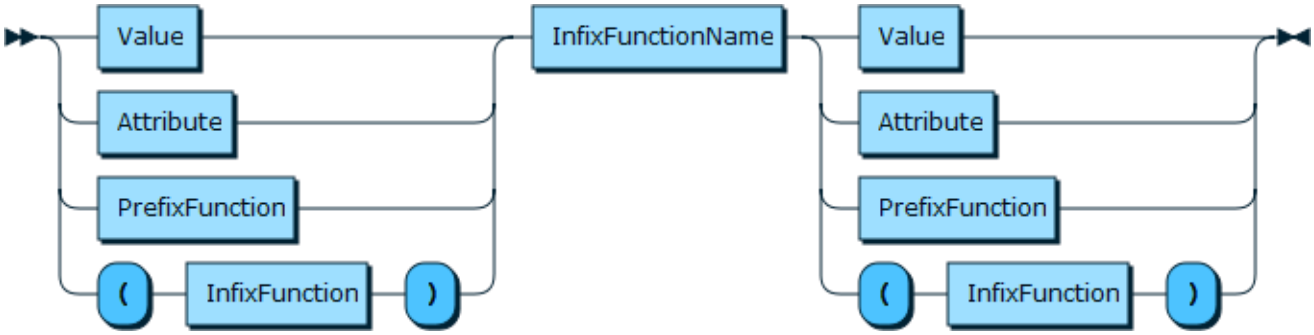
Anzahl	COUNT	Gibt die Anzahl der über die verschiedenen Objektinstanzen gesammelten Attribute mit gleichem Attributnamen zurück.

Diese Liste kann für weitere Präfix-Funktionen erweitert werden. Die genannten Funktionen müssen umgesetzt werden.

Die Parameter mehrstelliger Prefix-Funktionen sind in Anhang 8.3 zu dokumentieren/dokumentiert.

3.5 *InfixFunction*

InfixFunction:



Die Infix-Funktionen können Attribute (*Attribute*), feste Werte (Value*), Prefix-Funktionen (PrefixFunction) sowie geklammerte weitere Infix-Funktionen* mathematisch oder logisch über einen Operator mit Namen *InfixFunctionName* verknüpfen. Eine Klammerung für verschachtelte Infix-Funktionen ist zwingend vorgesehen. Regeln bzgl. Punkt vor Strich Rechnung entfallen somit.

Die Bezeichnung der Rückgabe erfolgt entweder entsprechend der Infix-Funktion oder bei Verwendung eines Alias-Namens (entsprechend der Definition in *AttributeFunctionList*) zur neu genannten Bezeichnung.

Mögliche Infix-Funktionen und ihre Infix*FunctionName*-Schlüsselworte/Zeichen sind in folgender, erweiterbaren Liste aufgeführt:

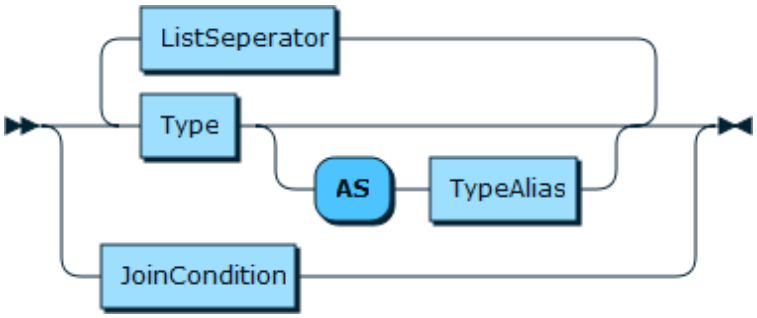
Infix-Funktion	**InfixFunctionName**	**Beschreibung**
(Klammerung)	(bzw.)	(Gruppiert mathematische oder logische Funktionen)
Addition	+	Gibt die Summe zweier (verschiedener) Attribute für jede Instanz der Liste der Objektinstanzen zurück
Subtraktion	-	Gibt die Differenz zweier (verschiedener) Attribute für jede Instanz der Liste der Objektinstanzen zurück
Multiplikation	*	Gibt die Multiplikation zweier (verschiedener) Attribute für jede Instanz der Liste der Objektinstanzen zurück
Division	/	Gibt die Division zweier (verschiedener) Attribute für jede Instanz der Liste der Objektinstanzen zurück
Modulo	%	Gibt den Modulo-Wert zweier (verschiedener) Attribute für jede Instanz der Liste der Objektinstanzen zurück
Bitweises Und	&	Gibt den bitweise „Und“ verknüpften Wert zurück

Bitweises Oder	|	Gibt den bitweise „Oder verknüpften Wert zurück
Bitweises Exklusiv-Oder	^	Gibt den bitweise „Exclusiv Oder“ verknüpften Wert zurück

Diese Liste kann für weitere Infix-Funktionen erweitert werden. Die genannten Funktionen müssen umgesetzt werden.

3.6 *TypesList - TypeAlias*

TypesList:



TypeAlias:

In der „FROM“-Klausel können ein oder mehrere Typen von Daten angegeben werden, aus deren Datenbestand die mittels „SELECT“ ausgewählten Attribute ausgegeben werden. Diese Typen werden als *Type* gegebenenfalls unter Verwendung des Listen-Seperators (*ListSeperator*, ein „Komma“) angegeben. Für diese Typen können auch jeweils, optional, neue Typnamen (*TypeAlias* beziehungsweise *TypeAliasName*) unter Verwendung des Schlüsselwortes „AS“ vergeben werden. Hierbei können diese neue Namen auch in „doppelten Anführungszeichen“ gesetzt werden, um auch Sonderzeichen verwenden zu können und eine etwaige mathematische Auswertung zu verhindern. Der *TypeAlias(Name)* ist die Bezeichnung unter der während der Anfrage, der Berechnung sowie bei der Ausgabe auf den ursprünglichen Typen referenziert wird. Die Verwendung neuer Namen erlaubt damit:

- Erstens eine erleichterte Schreibweise, damit nicht in jedem Teil der Anfrage-Zeichenkette ein unter Umständen länglicher Typname angegeben werden muss.
- Zweitens ein Self-Join.

Beide genannten Funktionen der Umbenennung sowie der Self-Join müssen umgesetzt werden.

Neben der einfachen Auswahl von Typen von Daten, können Datentypen auch mittels JOIN miteinander verknüpft werden (siehe nachfolgend *JoinCondition*).

3.7 *JoinCondition*

SQL bietet für die Verknüpfung mehrerer Tabellen verschiedene Joins an. Explizit als Join gibt es dort

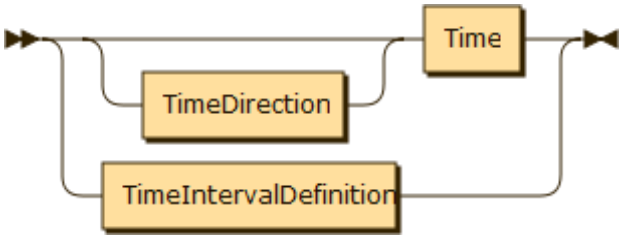
- INNER JOIN Inner Join,
- LEFT JOIN Left Join,
- RIGHT JOIN Right Join,
- FULL JOIN Full Join.

Hinzu kommt der implizite Self Join mittels Tabellen-Alias (s. auch *TypeList* beziehungsweise *TypeAlias*).

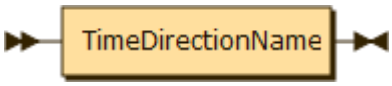
Die Joins und deren Verknüpung über eines oder mehrere Attribute sind wie in SQL auch in der Anfragesprache umzusetzen.

3.8 *TimeRestriction – TimeDirection*

TimeRestriction:



TimeDirection:



Zeiteinschränkungen werden in der Anfragesprache direkt hinter der Angabe der abzufragenden Typen der FROM Klauseln angegeben (siehe Definition von *SelectQuery*). Ohne die Zeiteinschränkung werden die ausgewählten (bzw. alle, falls keine weiteren Einschränkungen folgen) Zeitreihen komplett, vom erstem Zeitpunkt bis letzten Zeitpunkt gewählt. In der Zeiteinschränkung sind zwei Varianten der Abfrage möglich. Die Abfrage nach einem Zeitpunkt oder nach einem Zeitintervall. Die Abfrage nach Zeitpunkten erfolgt über die Angabe der Zeit (siehe nachfolgendes *Time*) mit optionaler Angabe der *TimeDirection*. Die *TimeDirection* gibt an, ob beispielsweise ein vorheriger oder ein nachfolgender Wert genommen werden soll, wenn zu exakt dem angegebenen Zeitpunkt kein Messwert übermittelt wurde.

Mögliche Stützstellenfunktion und ihre *TimeDirection*-Schlüsselworte sind in folgender, erweiterbaren Liste aufgeführt:

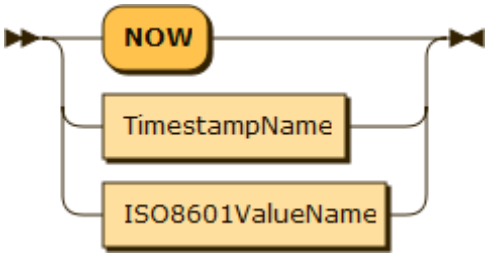
Stützstellenfunktion	**TimeDirection**	**Beschreibung**
Vorhergehender Wert	NEARESTBEFORE	Gibt den Wert mit nächstem Zeitstempel <= dem abgefragten Zeitpunkt zurück. [DEFAULT]
Nachfolgender Wert	NEARESTAFTER	Gibt den Wert mit nächstem Zeitstempel >= dem abgefragten Zeitpunkt zurück.

Diese Liste kann für weitere Stützstellenfunktionen erweitert werden. Die genannten Funktionen müssen umgesetzt werden. Sinnvoll ist es ein Default je Zeitreihe oder Zeitreihentyp in den Daten der Zeitreihen abzulegen und dies bei Anfragen automatisch zu Berücksichtigen. Insbesondere wird dadurch ermöglicht, Rechts und Linksgestempelte Zeitreihenwerte passend auch an Zeitpunkten abzufragen, an denen keine Stützstellen-Daten im Archiv vorliegen. Der Default-Fallback, falls weder eine *TimeDirection* angegeben ist, noch die Zeitreihen-spezifischen Defaults oder Zeitreihentyp-Defaults angegeben sind, ist „NEARESTBEFORE“, also der Wert mit dem nächsten Zeitstempel kleiner gleich dem abgefragten Zeitpunkt.

Für die Abfrage von Zeitintervallen siehe nachfolgende Definiton von *TimeIntervallDefinition*.

3.9 *Time – ISO8601ValueName*

Time:

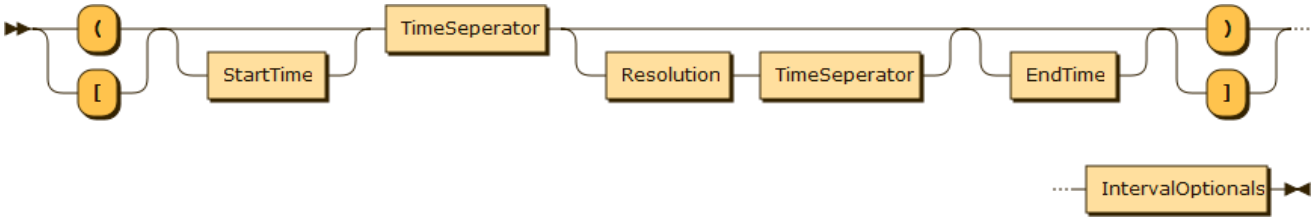


ISO8601ValueName:

In der Anfragesprache sind zwei Arten für die Angabe von Zeitpunkten möglich: Ein Zeitstempel (*TimestampName*) mit der Angabe der (SI-)Sekunden nach dem 1.1.1970 in UTC-Zeit ohne Schaltsekunden (siehe Unixzeit (The Open Group, 2018)). Ein Zeitpunkt in der ISO 8601-Notation (ISO, 1988) umgeben von den Schlüsselwörtern „ISO(„ und „)“. Desweiteren gibt es für die aktuelle Uhrzeit-Datum-Kombination das Schlüsselwort „NOW“.

3.10 *TimeIntervalDefinition – StartTime – EndTime*

TimeIntervalDefinition



StartTime:

EndTime:

Für die Angabe von Zeiträumen für Abfragen an Zeitreihenarchive wird in der Anfragesprache die *TimeIntervalDefinition* verwendet. Hierbei wird eine erweiterte mathematische Intervallnotation verwendet.

In dieser Notation wird ein Startzeitpunkt für die Rückgabewerte der Zeitreihen mit *StartTime* – gleich wie bei Zeitpunkten (siehe *Time)* – angegeben. Mit *EndTime* – gleich wie bei Zeitpunkten (siehe *Time*) – wird der Endzeitpunkt angegeben. Für die Anfrage werden alle Zeitreihendaten berücksichtigt, die zwischen Startzeitpunkt und Endzeitpunkt liegen – mit den nachfolgend vorgestellten Intervallgrenzen. Soll als Intervallstart der Beginn der Zeitreihe gewählt werden (-∞), so wird keine *StartTime* genannt. Das Feld bleibt leer. Soll das Intervallende zum Ende der in der Zeitreihe vorhandenen Zeitpunkte gewählt werden (∞), so wird keine *EndTime* genannt. Das Feld bleibt leer.

Die Intervallgrenzen für offene und halboffene Intervalle werden wie folgt notiert:

- Intervallgrenze am Intervallstart:
  - Eine öffnende Klammer „(„ für eine offene Intervallgrenze am Intervallstart. Der genannte Startwert *StartTime* wird also nicht mit einbezogen. Alle nachfolgenden Werte bis zum Intervallende *EndTime* werden berücksichtigt.
  - Eine öffnende Klammer „[„ für eine geschlossene Intervallgrenze am Intervallstart. Der genannte Startwert *StartTime* wird mit einbezogen. Alle Nachfolgenden Werte bis zum Intervallende *EndTime* werden berücksichtigt.
  - Bleibt die *StartTime* ungenannt, so ist es ohne Bedeutung, welche öffnende Klammer verwendet wird. Es wird von Beginn der Zeitreihe alle Werte bis zum Intervallende *EndTime* mit einbezogen
- Analog ist die Notation der Intervallgrenze am Intervallende definiert:
  - Eine schließende Klammer „)„ für eine offene Intervallgrenze am Intervallende. Der genannte Endwert *EndTime* wird also nicht mit einbezogen. Alle vorherigen Werte ab dem Intervallstart *StartTime* werden berücksichtigt.
  - Eine schließende Klammer „]„ für eine geschlossene Intervallgrenze am Intervallende. Der genannte Endwert *EndTime* wird mit einbezogen. Alle vorherigen Werte ab dem Intervallstart *StartTime* werden berücksichtigt.

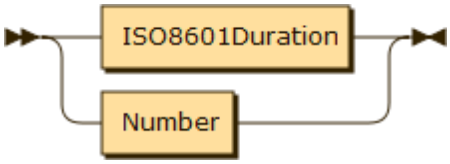
- Bleibt die *EndTime* ungenannt, so ist es ohne Bedeutung, welche öffnende Klammer verwendet wird. Es werden alle Werte ab dem Intervallstart *StartTime* bis zum Ende der Zeitreihe mit einbezogen.

Für die Trennung der Zeitangaben wird *TimeSeperator,* ein „Doppelpunkt“, verwendet.

Optional kann die zeitliche Auflösung mittels *Resolution* und weiterem *TimeSeperator* in der Mitte des Intervalls angegeben werden (siehe nachfolgend *Resolution*). Im Falle der Änderung der zeitlichen Auflösung, sind optional auch weitere Parameter für diese Änderung anzugeben (siehe nachfolgend IntervalOptionals). Wird die Zeitliche Auflösung nicht mittels *Resolution* angegeben, so sind etwaige angegebene *IntervalOptionals* nicht auszuwerten.

3.11 *Resolution*

Resolution:

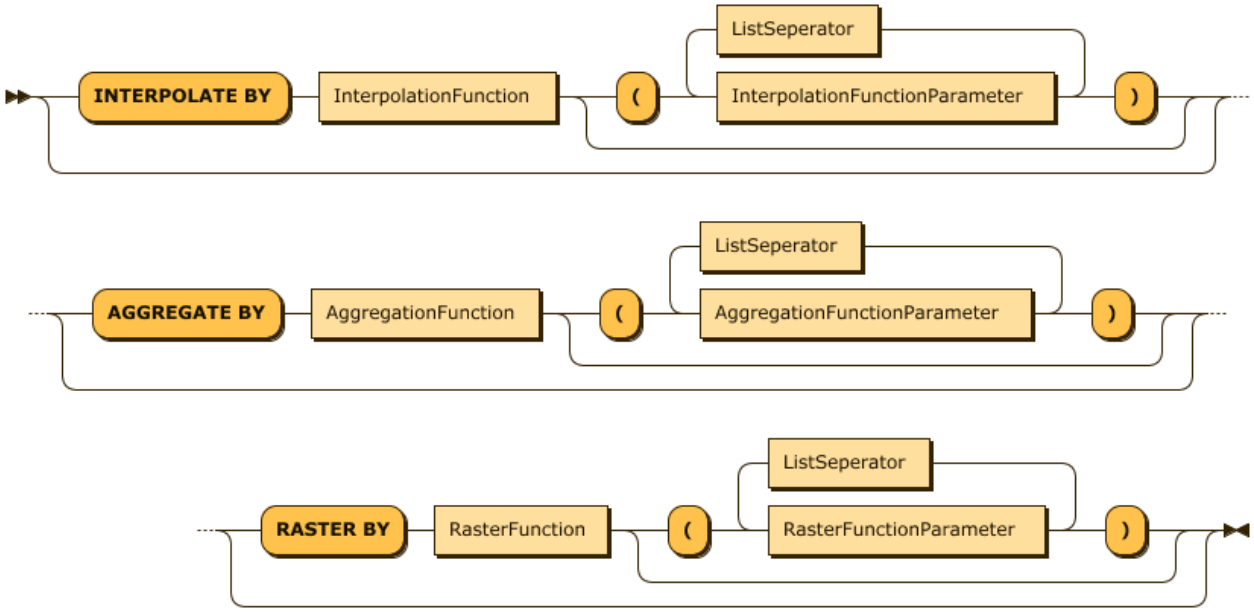


Die gewünschte zeitliche Auflösung von Zeitreihen kann für die Auswertung und Ausgabe mittels der Angabe der *Resolution* in der Zeitintervalldefinition *TimeIntervalDefintion* gewählt werden. Hier stehen zwei Möglichkeiten zur Wahl:

- Es wird mittels *Number* die Anzahl an Stützstellen der Ausgabe angegeben, welche auf das Intervall gleichverteilt werden (siehe auch nachfolgend *Raster*). Ist die Nummer < 1 kann keine Auswertung erfolgen. Ist die Nummer = 1, so wird der gesamte Zeitraum auf einen Wert gekürzt (siehe auch nachfolgend *AggregationFunction*).
- Es wird mittels der ISO 8601 Definition für Zeiträume (ISO, 1988) der zeitliche Abstand zwischen den Stützstellen der Ausgabe angegeben. Ist der angegebene Zeitraum länger als das Intervall, so ist keine Auswertung möglich. Ist der angegebene Zeitraum gleich dem Intervall, so wird der gesamte Zeitraum auf einen Wert gekürzt (siehe auch nachfolgend AggregationFunction).
- Sind Intervallstart oder Intervallende nicht genannt (-∞ bzw. +∞), so werden je Zeitreihe der Start- bzw. Endzeitpunkt der in der Zeitreihe im Archiv befindlichen Daten für die Auflösungsänderung verwendet.

3.12 *IntervalOptionals*

IntervalOptionals:



Wird in der *TimeIntervalDefinition* die zeitliche Auflösung mittels der Angabe einer Auflösung (siehe *Resolution*) geändert, so sind weitere Parameter für die Berechnung der neuen Stützstellen für die Auswertung und Ausgabe optional anzugeben. Diese Paremeter sind:

1. Die Interpolation. Welche Interpolation soll erfolgen, wenn die Auflösung der Zeitreihe in einem Bereich der Zeitreihe nicht ausreichend ist und keine Daten an den neuen Stützstellen vorhanden sind.
   Diese Angabe der Interpolation kann mittels des Schlüsselworts „INTERPOLATE BY“ eingeleitet werden. Es folgt der Bezeichner einer Interpolationsfunktion *InterpolationFunction* sowie optionale, listenseparierte (ListSeperator, ein „Komma“) Interpolationsparmeter *InterpolationFunctionParameter*. Die zur Verfügung stehenden Interpolationsfunktionen sind unter *InterpolationFunction* gelistet. Die Interpolationsparameter *InterpolationFunctionParameter* sind abhängig von der gewählten Interpolationsfunktion zu nennen. Sollten solche Parameter für die Berechnung der Interpolation nötig sein, so muss ein Parameter-Default existieren, der bei Ausbleiben der Angabe der Parameter für die Auswertung gewählt wird.
2. Die Aggregation. Welche Aggregation soll erfolgen, wenn die Auflösung der Zeitreihe in einem Bereich der Zeitreihe höher als die abgefragte Auflösung ist.
   Diese Angabe der Aggregation kann mittels des Schlüsselworts „AGGREGATE BY“ eingeleitet werden. Es folgt der Bezeichner einer Aggregationsfunktion *AggregationFunction* sowie optionale, listenseparierte (ListSeperator, ein „Komma“) Aggregationsparmeter *AggregationFunctionParameter*. Die zur Verfügung stehenden Aggregationsfunktionen sind unter *AggregationFunction* gelistet. Die Aggregationsparameter *AggregationFunctionParameter* sind abhängig von der gewählten Aggregationsfunktion zu nennen. Sollten solche Parameter für die Berechnung der Aggregation nötig sein, so muss ein Parameter-Default existieren, der bei Ausbleiben der Angabe der Parameter für die Auswertung gewählt wird.
3. Das Raster. Das Raster lässt eine zeitliche Verschiebung/Offset der in der Auswertung und für die Ausgabe berücksichtigten Abtastung in Abhängigkeit der Intervallgrenzen und der Auflösung zu. Die Angabe des Rasters kann mittels des Schlüsselworts „RASTER BY“ eingeleitet werden. Es folgt der Bezeichner einer Rasterfunktion *RasterFunction* sowie optionale, listenseparierte (ListSeperator, ein „Komma“) Rasterparmeter *RasterFunctionParameter*. Die zur Verfügung stehenden Rasterfunktionen sind unter *RasterFunction* gelistet. Die Rasterparameter *RasterFunctionParameter* sind abhängig von der gewählten Rasterfunktion zu nennen. Sollten solche Parameter für die Berechnung der Rasterung nötig sein, so muss ein Parameter-Default existieren, der bei Ausbleiben der Angabe der Parameter für die Auswertung gewählt wird.

3.13 *InterpolationFunction*

InterpolationFunction:

Mögliche Interpolationsfunktionen und ihre *InterpolationFunctionName*-Schlüsselworte sind in folgender, erweiterbaren Liste aufgeführt. Je Interpolationsfunktion in dieser Liste, findet sich eine weitere Liste mit den möglichen Interpolationsparametern *InterpolationFunctionParameter* im Anhang 8.2.1, **wenn Parameter nötig sind**. Eine Interpolation soll erfolgen, wenn die Auflösung der Zeitreihe in einem Bereich der Zeitreihe nicht ausreichend ist und keine Daten an den auszugebenden Stützstellen vorhanden sind. **Da nicht nur Zahlwerte in Zeitreihen abgelegt werden, sind auch für alle weitere Attributtypen dementsprechende Interpolationsmechanismen zu implementieren.**

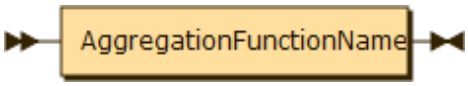
Interpolationsfunktion	InterpolationFunctionName	Beschreibung
Linksgestempelte Interpolation	STAMPED-LEFT	Gibt für Stützstellen den Wert mit dem nächstem Zeitstempel <= der abgefragten Stützstelle zurück. [DEFAULT]

Rechtsgestempelte Interpolation	STAMPED-RIGHT	Gibt für Stützstellen den Wert mit dem nächstem Zeitstempel >= der abgefragten Stützstelle zurück.
Lineare Interpolation	LINEAR	Gibt für Stützstellen den Wert mit der linearen Interpolation zwischen den Werte des nächst kleineren sowie des nächst größeren Zeitstempels aus.

Diese Liste ist beispielhaft und kann für weitere Interpolationsfunktionen dementsprechend erweitert werden. Die genannten Funktionen müssen umgesetzt werden. Sinnvoll ist es, ein Default je Zeitreihe oder Zeitreihentyp in den Daten der Zeitreihen abzulegen und dies bei Anfragen automatisch zu berücksichtigen. Insbesondere wird dadurch ermöglicht, rechts und linksgestempelte Zeitreihenwerte passend auch sinnvoll zu Zeitpunkten zu generieren, an denen keine Stützstellen-Daten im Archiv vorliegen. Der Default-Fallback, falls eine neue Auflösung (siehe *TimeIntervalDefinition* und *Resolution*) vorgegeben wird, jedoch weder eine Interpolationfunktion angegeben ist, noch die Zeitreihen-spezifischen Defaults oder Zeitreihentyp-Defaults angegeben sind, ist „STAMPED-LEFT.

3.14 *AggregationFunction*

AggregationFunction:



Mögliche Aggregationsfunktionen und ihre *AggregationFunctionName*-Schlüsselworte sind in folgender, erweiterbaren Liste aufgeführt. Je Aggregationsfunktion in dieser Liste, findet sich eine weitere Liste mit den möglichen Aggregationsparametern *AggregationFunctionParameter* im Anhang 8.2.2, **wenn Parameter nötig sind**. Eine Aggregation soll erfolgen, wenn die Auflösung der Zeitreihe in einem Bereich der Zeitreihe höher als die abgefragte Auflösung ist. **Da nicht nur Zahlwerte in Zeitreihen abgelegt werden, sind auch für alle weitere Attributtypen dementsprechende Aggregationsmechanismen zu implementieren.**

Aggregationsfunktion	**AggregationFunctionName**	**Beschreibung**
Durchschnitt	AVG	Gibt den Mittelwert aller im Zeitraum **vorkommenden** Werte zeitunabhängig zurück. [DEFAULT]
Durchschnitt Linksgestempelt	AVGSL	Gibt den Mittelwert aller im Zeitraum **gültigen** Werte **zeitgewichtet** zurück. Gültig sind bei linksgestempelten Werten die Werte die direkt vor (oder auf – falls existent) dem Intervallstart beginnen und direkt vor dem Intervallende enden.
Durchschnitt Rechtsgestempelt	AVGSR	Gibt den Mittelwert aller im Zeitraum **gültigen** Werte **zeitgewichtet** zurück. Gültig sind bei rechtsgestempelten Werten die Werte die nach dem Intervallstart beginnen und auf (falls existent) oder direkt nach dem Intervallende enden.
Minimum	MIN	Gibt den Minimalwert aller gemessenen Werte im Zeitraum zurück.
Maximum	MAX	Gibt den Maximalwert aller gemessenen Werte im Zeitraum zurück.

Median	MEDIAN	Gibt den Median der gemessenen Werte im Zeitraum zurück

Diese Liste ist beispielhaft und kann für weitere Aggregationfunktionen dementsprechend erweitert werden. Die genannten Funktionen müssen umgesetzt werden. Sinnvoll ist es, ein Default je Zeitreihe oder Zeitreihentyp in den Daten der Zeitreihen abzulegen und dies bei Anfragen automatisch zu berücksichtigen. Insbesondere wird dadurch ermöglicht, Werte mehrerer Stützstellen in Abhängigkeit der darin integrierten Zeitreihendaten sinnvoll zusammenzufassen – je nach Vorhaben können jedoch andere Aggregationen sinnvoll sein. Der Default-Fallback, falls eine neue Auflösung (siehe *TimeIntervalDefinition* und *Resolution*) vorgegeben wird, jedoch weder eine Aggregationsfunktion angegeben ist, noch die Zeitreihen-spezifischen Defaults oder Zeitreihentyp-Defaults angegeben sind, ist „AVG“.

3.15 *RasterFunction*

RasterFunction:

Mögliche Rasterfunktionen und ihre *RasterFunctionName*-Schlüsselworte sind in folgender, erweiterbaren Liste aufgeführt. Je Rasterfunktion in dieser Liste, findet sich eine weitere Liste mit den möglichen Rasterparametern *RasterFunctionParameter* im Anhang 8.2.3, **wenn Parameter nötig sind**. Die Rasterung lässt eine zeitliche Verschiebung/Offset der in der Auswertung für die Ausgabe berücksichtigten Abtastung in Abhängigkeit der Intervallgrenzen und der Auflösung zu. **Da nicht nur Zahlwerte in Zeitreihen abgelegt werden, sind auch für alle weitere Attributtypen dementsprechende Rastermechanismen zu implementieren.**

Aggregationsfunktion	**AggregationFunctionName**	**Beschreibung**
Strikte Intervallgrenzen	STRICT	Verarbeitet ein Raster mit exakt den im Zeitintervall genannten Zeitpunkten. [DEFAULT]
Gerade Uhrzeiten	EVEN	Verschiebt das Raster auf „gerade Uhrzeiten“ innerhalb des genannten Zeitintervalls.

Diese Liste ist beispielhaft und kann für weitere Rasterfunktionen dementsprechend erweitert werden. Die genannten Funktionen müssen implementiert werden. Sinnvoll ist es, ein Default je Zeitreihe oder Zeitreihentyp in den Daten der Zeitreihen abzulegen und dies bei Anfragen automatisch zu berücksichtigen. Der Default-Fallback, falls eine neue Auflösung (siehe *TimeIntervalDefinition* und *Resolution*) vorgegeben wird, jedoch weder eine Rasterfunktion angegeben ist, noch die Zeitreihen-spezifischen Defaults oder Zeitreihentyp-Defaults angegeben sind, ist „STRICT“.

3.16 *QueryConditions*

Die QueryConditions werden unverändert von SQL übernommen und hier gegebenenfalls unvollständig aufgezählt – **sind aber vollständig zu implementieren**. Die vollständigen Definitonen sind der SQL Standard-Definition zu entnehmen. Beispiele dazu finden sich online unter W3School-SQL (Refsnes Data, 2018).

3.16.1 WHERE

SQL bietet für die Auswahl von Daten anhand ihrer Werte die WHERE-Klausel an, um Bedingungen an die Daten einzubringen.

Dort gibt es

- Arithmetische und Bitweise-Verknüpfungen
    - Wie die gelisteten Infix-Funktionen (siehe *InfixFunction*)

- Logische Verknüpfungen
  - AND Und
  - OR Oder
  - NOT Nicht
  - ANY/SOME Wahr, wenn eines der Unterabfragenwerte die Bedingung erfüllt
  - ALL Wahr, wenn alle Unterabfragenwerte die Bedingungen erfüllen
  - EXISTS Wahr, wenn die Unterabfrage einen oder mehrere Werte zurückgibt
  - ( ) Klammerung
- Einfache Vergleichsoperatoren
  - = Gleich
  - <> Ungleich
  - > Größer
  - < Kleiner
  - >= Größer oder gleich
  - <= Kleiner oder gleich
- Weitere Vergleichsoperatoren
  - BETWEEN AND Zwischen einem Bereich inklusive dem Rand des Bereichs
  - LIKE Pattern Suche nach einem Pattern
    - Wildcard Patterns
      - % Repräsentiert keines, eines oder mehrere Zeichen
      - _ Repräsentiert ein einzelnes Zeichen
  - IN ( ) Zur Spezifikation multipler möglicher Werte für ein Attribut
    - Liste
    - SELECT Unterabfrage
  - IS (NOT) NULL Test auf (nicht) NULL Werte, also (nicht) leere Felder. Wert 0 oder „“ sind nicht NULL.
- Numerische, logische, zeitbezogene Infix- oder Prefix-Operatoren

Alle beschriebenen WHERE-Klausel-Möglichkeiten müssen umgesetzt werden. Desweiteren sind alle beschriebenen Prefix-Funktionen (siehe *PrefixFunction*) für die Verwendung in WHERE zu implementieren.

3.16.2 GROUP BY

SQL bietet für die Gruppierung von Daten die GROUP BY-Klausel an, welche auch in der Anfragesprache umgesetzt werden muss. Für die Verwendung in GROUP BY sind auch alle beschriebenen Prefix-Funktionen (siehe *PrefixFunction*) zu implementieren.

3.16.3 HAVING

SQL bietet mit der HAVING-Klausel die Möglichkeit in den *QueryConditions*-Teil auch Aggregationsfunktionen mit einzubringen. Diese Klausel ist für die Anfragesprache zu übernehmen. Für die Verwendung in HAVING sind auch alle beschriebenen Prefix-Funktionen (siehe *PrefixFunction*) umzusetzen. Darüber hinaus sind auch alle Möglichkeiten der WHERE-Klausel analog in die HAVING-Klausel der Anfragesprache zu überführen und umzusetzen.

3.16.4 ORDER

SQL bietet mit der ORDER BY-Klausel die Möglichkeit die Ausgabe aufsteigend („ASC“) oder absteigend („DESC“) zu sortieren. Dies ist auch für die Anfragesprache – auch für mehrere angegebene Sortierungen – wie in SQL zu implementieren. Ohne ORDER BY-Klausel müssen Ergebnisse der Anfragesprache aufsteigend zeitlich sortiert ausgegeben werden.

3.16.5 UNION und UNION ALL

SQL bietet mit dem UNION- beziehungsweise UNION ALL-Operator die Möglichkeit, das Ergebnis zweier oder mehrerer Anfragen zu kombinieren. Dies ist auch in der Implementierung der Anfragesprache umzusetzen.

3.16.6 Kommentare

SQL bietet mit der „—„-Zeichenfolge zeilenweise Kommentare beziehungsweise mit „/*“ „*/“ zeilenübergreifende Kommentare anzugeben. Die gleiche Funktionalität soll auch für die Anfragesprache implementiert werden.

3.16.7 SQL Prefix-Funktionen

Desweiteren bieten die diversen SQL Implementierungen zum Teil unterschiedliche Funktionen an, die nach Bedarf zusätzlich in die Anfragesprache integriert werden können – jedoch nicht zwingend erforderlich sind.

- String Funktionen (CONCAT, LENGTH, INSTR,LOWER,UPPER…)
- Numerische Funktionen (ABS, SIN, COS, ROUND, COUNT, MIN, MAX,
- Zeit/Datums-Funktionen (ADD_MONTHS, CURRENT_DATE, LAST_DAY, LOCALTIMESTAMP,…)
- Erweiterte Funktionen (BIN, COALESCE, IF, CONVERT,…)
- Diverse SQL Implementierung bieten die Möglichkeit NULL-Werte bei Berechnungen z.B. mittels des Schlüsselwortes „IFNULL“ (MySQL) zu ersetzen.

4 Rückgabe

Die Rückgabe der Anfragesprache unterliegt grundsätzlich einer Tabellenstruktur. Die Rückgabe erfolgt als comma-seperated values (CSV) nach RFC 4180 (Shafranovich, 2005). Wesentliche Inhalte der RFC 4180:

- Trennzeichen ist das Zeichen „Komma“,
- Informationen die ein Komma, ein Zeilenvorschub oder doppelte Anführungszeichen enthalten, sollten in doppelten Anführungszeichen geschrieben werden,
- In Informationen die doppelte Anführungszeichen enthalten wird jedem doppeltem Anführungszeichen ein weiteres doppeltes Anführungszeichen vorangestellt.

Der in der RFC 4180 optionale Header Eintrag muss explizit für die Rückgabe befüllt werden: Die erste Zeile der Rückgabe-CSV erhält die Bezeichner aus der *AttributeFunctionList* beziehungsweise bei Verwendung des „*“-Operators die vorkommenden Attributnamen *AttributeName* des ausgewählten Typen *Type.* Siehe Kapitel 3 für die Beschreibung der Syntax und Semantik der kursiv dargestellten Bausteine der Anfragesprache. Die Ausgabe erfolgt in der in der Anfrage aufgeführten Reihenfolge, beziehungsweise bei Verwendung des „*“-Operators in beliebiger, aber für die Ausgabe feststehender Reihenfolge. Werden Funktionen (*InfixFunction*/*PrefixFunction*) oder Aliase (*AttributeAliasName*) auf Attribute angewendet, so wird die entsprechende Funktionsbeschreibung oder der Alias in dem Feld des CSV-Header genannt.

Es erfolgt nur die Rückgabe der in der *AttributeFunctionList* aufgelisteten Attribute oder – bei Verwendung des „*“-Operators – die im Archiv unter den in der *TypeList* aufgelisteten Typen verfügbaren Attribute.

5 Beispiele

Im Folgenden werden Beispiele für Abfragen an Zeitreihenarchiven mittels der Anfragesprache beschrieben.

5.1 *Zeitreihen-Typen*

Eine Datenbank enthält Zeitreihen einer oder mehrerer Typen. Jeder Typ wird durch einen Namen (zum Beispiel „SmartMeter“ oder „SCADAMeasurement“) identifiziert. Die Zeitreihen enthalten Zeitreiheneinträge mit zu Attributen zugehörigen Daten. Zur Identifizierung von Zeitreiheneinträgen werden der Zeitreihentyp (type), der Zeitreihengeräte-Identifier (mrid), der Zeitstempel (timestamp) sowie die (unit) beziehungsweise der Messtyp (measurementType) herangezogen.

Nachfolgend findet sich eine Auswahl von Zeitreiheneinträgen vom Typ SmartMeter:

Tabelle 1: SmartMeter Zeitreihe

type	mrid	timestamp	value	unit	Multiplier	category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm1	1537805700	17358.7	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	
SmartMeter	Sm2	1537805700	2315.1	Wh	K	SubstitutedValue

Das Beispiel enthält acht Zeitreiheneinträge vom Typ SmartMeter. Die Zeitreiheneinträge sind eindeutig zu indentifizieren über type, mrid, timestamp und unit. Für zwei unterschiedliche gelistete Smart Meter (Sm1 und Sm2 mit fester unit Wh und type SmartMeter) gibt es jeweils 4 Zeitreiheneinträge mit unterschiedlichem Zeitpunkt (Werte des Attributs timestamp) und Werten der Attribute value, multiplier, category.

5.2 *SELECT Statement*

Das SELECT Statement wird verwendet, um Daten abzufragen. Die Daten werden in einer Ausgabetabelle ausgegeben. Die folgenden Beispiele beziehen sich auf die Zeitreihe in Tabelle 1: SmartMeter Zeitreihe.

5.2.1 SELECT AttributeName Beispiel:

Die folgende Anfrage wählt die „mrid“ und „timestamp“ der Zeitreihen vom Typ SmartMeter ohne Zeiteinschränkung, gibt also zurück, zu welchem Smart Meter zu welchen Zeitpunkten Zeitreiheneinträge abgelegt sind.

```
SELECT mrid, timestamp FROM SmartMeter;
```

Das Ergebnis sieht wie folgt aus:

mrid	timestamp
Sm1	1537803000
Sm2	1537803000
Sm1	1537803900
Sm2	1537803900
Sm1	1537804800
Sm2	1537804800

Sm1	1537805700
Sm2	1537805700

Die Sortierung erfolgt aufsteigend anhand der Zeitstempel, soweit keine andere Sortierung vorgegeben ist (siehe Abschnitt 5.10.2)

5.2.2 SELECT * Beispiel:

Die folgende Anfrage wählt alle Attribute der Zeitreihen vom Typ SmartMeter ohne Zeiteinschränkung

```
SELECT * FROM SmartMeter;
```

Das Ergebnis sieht wie folgt aus:

type	mrid	timestamp	value	unit	multiplier	category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	
SmartMeter	Sm1	1537805700	17358.7	Wh	K	
SmartMeter	Sm2	1537805700	2315.1	Wh	K	SubstitutedValue

Solche Anfragen sind mit Vorsicht einzusetzen, da im Archiv unter Umständen sehr viele korrespondierende Daten vorhanden sind und die Ausgabe dementsprechend groß wird.

5.3 *SELECT DISTINCT Statement*

Das SELECT DISTINCT Statement wird verwendet, um sich unterscheidende Daten abzufragen. Nur sich unterscheidende Daten werden in einer Ausgabetabelle ausgegeben, Dopplungen weggelassen. Das folgende Beispiel bezieht sich auf die Zeitreihe in Tabelle 1: SmartMeter Zeitreihe.

5.3.1 DISTINCT AttributeName Beispiel:

Die folgende Anfrage wählt die sich unterscheidenden „mrid“ der Zeitreihen vom Typ SmartMeter ohne Zeiteinschränkung (siehe dazu Abschnitte 5.7f), gibt also zurück, zu welchem Smart Meter jemals Daten abgelegt wurden.

```
SELECT DISTINCT mrid FROM SmartMeter;
```

Das Ergebnis sieht wie folgt aus:

Mrid
Sm1
Sm2

5.4 *LIMIT Statement*

Das LIMIT Statement wird verwendet, um die Anzahl der Zeitreiheneinträge in der Ausgabe auf die angegebene Anzahl zu verringern – falls mehr als die angebenene Anhahl an Ergebnisse vorliegen. Die Daten werden in einer Ausgabetabelle ausgegeben. Das folgende Beispiel bezieht sich auf die Zeitreihe in Tabelle 1: SmartMeter Zeitreihe.

5.4.1 LIMIT Beispiel:

Die folgende Anfrage wählt alle „mrid“ der Zeitreihen vom Typ SmartMeter ohne Zeiteinschränkung, und limitiert die Ausgabe auf einen Zeitreiheneintrag.

```
SELECT mrid FROM SmartMeter LIMIT 1;
```

Ein mögliches Ergebnis sieht wie folgt aus:

Mrid
Sm1

Es könnte hier aber auch Sm2 hier genannt werden, was abhängig von der Implementierung (und der Zeitstempel der abgelegten Daten) sein kann.

5.5 *SELECT Attribute-Alias*

Im SELECT können Attributnamen per AS umbenannt werden, sodass sowohl die innere Auswertung als auch die Ausgabe anhand der neuen Attribute erfolgt. Das Beispiel bezieht sich auf die Zeitreihe in Tabelle 1: SmartMeter Zeitreihe.

5.5.1 Attribute AS Beispiel:

Die folgende Anfrage vergibt mittels AS für das Attribut timestamp einen neuen Namen ts:

```
SELECT mrid, timestamp AS ts FROM SmartMeter
```

Die Rückgabe erscheint wie folgt:

mrid	**ts**
Sm1	1537803000
Sm2	1537803000
Sm1	1537803900
Sm2	1537803900
Sm1	1537804800
Sm2	1537804800
Sm1	1537805700
Sm2	1537805700

5.6 *SELECT multipler Typen mit Typ-Alias bzw. JOINs*

Das SELECT Statement kann auch über mehrere Typen angewendet werden. Diese Typen können mittels für die Auswertung und Ausgabe umbenannt werden. Das folgende Beispiel bezieht sich auf die Tabelle 1: SmartMeter Zeitreihe und die folgende Zeitreihe vom Typ SM_Plausibility:

Tabelle 2: SM_Plausibitlity Zeitreihe

type	**mrid**	**timestamp**	**unit**	**plausibilty_value**	**plausibiltiy_timestamp**	**plausibility_source**
SM_Plausibility	Sm1	1537804800	Wh	0.75	1537804850	Historical
SM_Plausibility	Sm1	1537804800	Wh	0.4	1537804860	Weather
SM_Plausibilty	Sm2	1537804800	W	0.8	1537804880	Historical

5.6.1 Types AS Beispiel:

Die folgende Anfrage wählt mehrere Attribute mehrerer Zeitreihen und gibt diese getrennt zurück:

```
SELECT SM.mrid AS mrid, SM.timestamp AS ts, SM.value AS value, SM.unit AS “sm-unit”,
PL.unit as “pl-unit”, PL.plausibility_value AS “pl-value”, PL.plausibility_source AS
“pl-source” FROM SmartMeter AS SM, SM_Plausibility AS PL
```

Die Rückgabe erscheint wie folgt:

mrid	ts	value	sm-unit	pl-unit	pl-value	pl-source
Sm1	1537803000	17350.2	Wh			
Sm2	1537803000	2301.4	Wh			
Sm1	1537803900	17351.3	Wh			
Sm2	1537803900	2306.8	Wh			
Sm1	1537804800	17355.0	Wh			
				Wh	0,75	Historical
				Wh	0,4	Weather
				W	0.8	Historical
Sm2	1537804800	2312.2	Wh			
Sm1	1537805700	17358.7	Wh			
Sm2	1537805700	2315.1	Wh			

Die Ausgabe ist nur bedingt verwendbar, da die zwei Tabellen zwar ineinander geschrieben werden, aber der Bezug zwischen den mrids, den Zeitstempeln und den Einheiten nicht hergestellt wird (siehe dazu JOIN in Abschnitt 5.10). Die Reihenfolge der Ausgabe erfolgt wie bisher aufsteigend sortiert nach Zeitstempel, da keine andere Angabe gemacht wurde (siehe dazu Abschnitt 5.10.2). Die Nennung der drei Zeitreiheneinträge vom Typ SM_Plausibility zwischen den entsprechenden Zeitreiheneinträgen von SmartMeter Sm1 und SmartMeter Sm2 zum Zeitpunkt 1537804800 ist zufällig – die Default-Sortierung erfolgt nur anhand der timestamps. Zu beachten ist weiterhin, dass die Alias-Ausdrücke teilweise mit „doppelten Anführungszeichen“ umgeben sind, um eine mathematische Auswertung des Ausdrucks zu verhinden.

5.6.2 JOINS

Mittels JOINS können Werte in Zeitreihen unterschiedlichen Typs verknüpft werden. Das folgende Beispiel zeigt eine Möglichkeit des JOINS. Weitere Möglichkeiten sind in der W3School für SQL (Refsnes Data, 2018) einzusehen.

Als Datengrundlage wird hier die Tabelle 1: SmartMeter Zeitreihe sowie Tabelle 2: SM_Plausibitlity Zeitreihe verwendet.

Die Zeitreihen sollen so verknüpft werden, dass für einen Zeitreiheneintrag der Zeitreihe SmartMeter die Plausibilitäten sowie deren Quellen mit angegeben werden sollen. Die Anfrage ist analog zu Abschnitt 5.6.1 definiert. Mittels LEFT JOIN werden alle Zeitreiheneinträge des ersten Typs aufgelistet, und um die Informationen des zweiten Types ergänzt, für die die ON-Bedingung gilt.

Die Anfrage lautet:

```
SELECT SM.mrid AS mrid, SM.timestamp AS ts, SM.value AS value, SM.unit AS unit,
PL.plausibility_value AS “pl-value”, PL.plausibility_source AS “pl-source”
FROM SmartMeter AS SM
LEFT JOIN SM_Plausibility AS PL ON SM.mrid=PL.mrid AND SM.timestamp=PL.timestamp AND
SM.unit=PL.unit
```

Die Rückgabe erscheint wie folgt:

mrid	ts	value	unit	pl-value	pl-source
Sm1	1537803000	17350.2	Wh		
Sm2	1537803000	2301.4	Wh		

Sm1	1537803900	17351.3	Wh		
Sm2	1537803900	2306.8	Wh		
Sm1	1537804800	17355.0	Wh	0,75	Historical
Sm1	1537804800	17355.0	Wh	0,4	Weather
Sm2	1537804800	2312.2	Wh		
Sm1	1537805700	17358.7	Wh		
Sm2	1537805700	2315.1	Wh		

Der Eintrag zum Sm1 zum Zeitpunkt 1537804800 wird doppelt dargestellt, da 2 Plausibilitäten zur Verfügung stehen. Der Eintrag zum Sm2 zum Zeitpunkt 1537804800 wird nicht erweitert, da die Einheiten in beiden Zeitreihen unterschiedlich (Wh vs. W) sind und deshalb die ON-Bedingung nicht greift.

5.7 *SELECT mit Zeitpunktangabe*

Mittels der Zeitpunktangabe können Werte zu festen Zeitpunkten abgefragt werden. Die folgenden Beispiele beziehen sich auf die Tabelle 1: SmartMeter Zeitreihe. Alle bisherigen Anfragenbeispiele sind mit Vorsicht einzusetzen, da im Archiv unter Umständen sehr viele korrespondierende Daten vorhanden sind und die Ausgabe dementsprechend groß wird, wenn der Suchraum nicht durch Zeitpunkte, Zeitintervalle (siehe Abschnitt 5.8) oder weitere Bedingungen (siehe Abschnitte 5.10ff) eingeschränkt wird.

5.7.1 Zeitpunktangabe für linksgestempelte Werte

Wird eine Zeitpunktangabe gemacht, wird automatisch der zu dem Zeitpunkt linksgestempelt gültige Wert verwendet.

```
SELECT mrid, timestamp FROM SmartMeter 1537803200
```

Die Rückgabe listet die SmartMeter mrids und deren Aufnahmezeitpunkt, welche zuletzt vor, oder auf dem abgefragten Zeitpunkt mit Zeitstempel 1537803200 gültig sind.

mrid	**timestamp**
Sm1	1537803000
Sm2	1537803000

Gleiches gilt für die Angabe mit der expliziten Verwendung des Defaults

```
SELECT mrid, timestamp FROM SmartMeter NEARESTBEFORE 1537803200
```

5.7.2 Zeitpunktangabe NOW

Als besonderes Schlüsselwort kann NOW für den aktuellen Zeitpunkt verwendet werden. Im folgenden Beispiel entspricht der aktuelle Zeitstempel dem Wert 1537803200 (s.o.)

```
SELECT mrid, timestamp FROM SmartMeter NOW
```

Auch hier ist die Angabe des Defaults optional Die Rückgabe listet die SmartMeter mrids und deren Aufnahmezeitpunkt, welche zuletzt vor, oder auf dem abgefragten Zeitpunkt NOW (mit Zeitstempel 1537803200) gültig sind.

mrid	**timestamp**
Sm1	1537803000
Sm2	1537803000

Gleiches gilt für die Angabe mit der expliziten Verwendung des Defaults

```
SELECT mrid, timestamp FROM SmartMeter NEARESTBEFORE NOW
```

5.7.3 Zeitpunktangabe nach ISO 8601

Die Zeitangabe kann auch als ISO 8601 Wert erfolgen. Der Timestamp 1537803000 entspricht dem Wert 2018-09-24T15:30:00+00:00

```
SELECT mrid, timestamp FROM SmartMeter ISO(2018-09-24T15:30:00+00:00)
```

Auch hier ist die Angabe des Defaults optional Die Rückgabe listet die SmartMeter mrids und deren Aufnahmezeitpunkt, welche zuletzt vor, oder auf dem abgefragten Zeitpunkt NOW (mit Zeitstempel 1537803000) gültig sind.

mrid	timestamp
Sm1	1537803000
Sm2	1537803000

Gleiches gilt für die Angabe mit der expliziten Verwendung des Defaults

```
SELECT mrid, timestamp FROM SmartMeter NEARESTBEFORE ISO(2018-09-24T15:30:00+00:00)
```

5.7.4 Zeitpunktangabe für rechtsgestempelte Werte

Um rechtsgestempelte Werte abfragen zu können, kann das Schlüsselwort NEARESTAFTER verwendet werden

```
SELECT mrid, timestamp FROM SmartMeter NEARESTAFTER NOW
```

Auch hier ist die Angabe des Defaults optional Die Rückgabe listet die SmartMeter mrids und deren Aufnahmezeitpunkt, welche zuletzt vor, oder auf dem abgefragten Zeitpunkt NOW (mit Zeitstempel 1537803200) gültig sind.

mrid	timestamp
Sm1	1537803900
Sm2	1537803900

5.8 *SELECT mit Zeitintervallangabe*

Mittels der Zeitintervallangabe können Werte zu Zeitintervallen abgefragt werden. Da im Archiv unter Umständen sehr viele, zu einer Anfrage korrespondierende Daten vorhanden sind und die Ausgabe dementsprechend groß wird, sollte der Suchraum durch Zeitpunkte, Zeitintervalle (siehe Abschnitt 5.8) oder weitere Bedingungen (siehe Abschnitte 5.10ff) eingeschränkt werden. Die folgenden Beispiele beziehen sich auf die Tabelle 1: SmartMeter Zeitreihe.

5.8.1 Geschlossene Zeitintervallangabe [ ]

Sollen genannte Zeitwerte am Rand der Intervalle mit in die Auswertung einfließen, ist es möglich, eine geschlossenes Intervall abzufragen:

```
SELECT * FROM SmartMeter [1537803000 : 1537804900]
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche im genannten Zeitraum beziehungsweise auf den Grenzen liegen:

type	mrid	timestamp	Value	unit	multiplier	category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	

5.8.2 Offene Zeitintervallangabe ( )

Sollen genannte Zeitwerte am Rand der Intervalle nicht mit in die Auswertung einfließen, ist es möglich, eine offenes Intervall abzufragen:

```
SELECT * FROM SmartMeter (1537803000 : 1537804900)
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche innerhalb des genannten Zeitraumes ohne die Grenzen liegen:

type	mrid	timestamp	value	Unit	multiplier	category
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	

5.8.3 Halboffene Zeitintervallangabe [ ) oder ( ]

Halboffene Zeitintervalle sind ebenfalls möglich. Die Anfragen

```
SELECT * FROM SmartMeter [1537803100 : 1537804900)
```

beziehungsweise

```
SELECT * FROM SmartMeter (1537803000 : 1537804800]
```

generieren das gleiche Ergebnis wie zuvor:

type	mrid	timestamp	value	Unit	multiplier	category
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	

5.8.4 Intervall von Zeitreihenanfang

Durch Weglassen des Startzeitpunktes im Intervall, wird der Beginn der Zeitreihe als Intervallstart gewählt. Das Ende des Intervalls definiert sich wie bisher.

```
SELECT * FROM SmartMeter ( : 1537804800)
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche von Beginn der Zeitreihen bis zum Zeitstempel 1537804800 (ausgenommen) liegen:

Type	mrid	timestamp	Value	unit	multiplier	category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	

Welches Klammerzeichen am Intervallstart verwendet wird, ist in diesem Fall ohne Belang. Das gleiche Ergebnis wird für folgende Abfrage geliefert:

```
SELECT * FROM SmartMeter [ : 1537804800)
```

5.8.5 Intervall bis Zeitreihenende

Durch Weglassen des Endzeitpunktes im Intervall, wird das Ende der Zeitreihe als Intervallende gewählt. Der Start des Intervalls definiert sich wie bisher.

```
SELECT * FROM SmartMeter [1537803900 : ]
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche vom Zeitstempel 1537803900 (eingeschlossen) bis zum Ende der Zeitreihen liegen:

Type	mrid	timestamp	value	unit	multiplier	category
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	
SmartMeter	Sm1	1537805700	17358.7	Wh	K	
SmartMeter	Sm2	1537805700	2315.1	Wh	K	SubstitutedValue

Welches Klammerzeichen am Intervallende verwendet wird, ist in diesem Fall ohne Belang. Das gleiche Ergebnis wird für folgende Abfrage geliefert:

```
SELECT * FROM SmartMeter [1537803900 : )
```

5.8.6 Intervall von Zeitreihenstart bis Zeitreihenende

Folgende Notationen sind äquivalent und liefern dasselbe Ergebnis (siehe dazu Abschnitt 5.2.2).

```
SELECT * FROM SmartMeter [ : ]
SELECT * FROM SmartMeter ( : ]
SELECT * FROM SmartMeter [ : )
SELECT * FROM SmartMeter [ : ]
SELECT * FROM SmartMeter
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter von Zeitreihenstart bis Zeitreihenende:

type	mrid	timestamp	value	unit	multiplier	category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm2	1537804800	2312.2	Wh	K	
SmartMeter	Sm1	1537805700	17358.7	Wh	K	
SmartMeter	Sm2	1537805700	2315.1	Wh	K	SubstitutedValue

5.9 *SELECT mit Zeitintervallangabe und Auflösungsveränderung*

Mittels der Zeitintervallangabe können Werte zu Zeitintervallen abgefragt werden. Die folgenden Beispiele beziehen sich auf die folgende Tabelle (hier nur unregelmäßige SmartMeter Sm1 Zeitreiheneinträge):

Zur Berechnung der Veränderung der Auflösung sind Interpolations-, Aggregations- und Rasterfunktionen nötig. Werden diese nicht angegeben, wird jeweils ein Default-Verfahren verwendet. In den folgenden Beispielen sind - falls nicht anders beschrieben - die Default-Verfahren zur Anwendung gekommen.

Tabelle 3: SmartMeter Sm1 mit unterschiedlichen Zeitabständen

type	mrid	timestamp	value	unit	multiplier	Category
SmartMeter	Sm1	1537802775	17350.0	Wh	K	
SmartMeter	Sm1	1537803000	17350.2	Wh	K	

SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm1	1537804125	17352.5	Wh	K	
SmartMeter	Sm1	1537805700	17358.7	Wh	K	

5.9.1 Auflösungsveränderung mit Interpolation

Wird eine Zeitreihe mit einer neuen Auflösung abgefragt, so wird eine Interpolation durchgeführt, wenn die abgefragte Auflösung höher als die der Datengrundlage ist. Die Auflösung wird in der Intervallmitte angegeben und das Interpolationsverfahren hinter dem Intervall mit dem Schlüsselwort INTERPOLATE BY gewählt. Die Liste möglicher Interpolationsverfahren und ihre Auswirkung ist Abschnitt 3.13 zu entnehmen.

```
SELECT * FROM SmartMeter [1537803000 : 5 : 1537803900] INTERPOLATE BY STAMPED-LEFT
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche im genannten Zeitraum beziehungsweise auf den Grenzen liegen und interpoliert mittels des Verfahrens für Links-gestempelte Werte STAMPED-LEFT:

type	mrid	timestamp	value	unit	multiplier	Category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm1	1537803225	17350.2	Wh	K	
SmartMeter	Sm1	1537803450	17350.2	Wh	K	
SmartMeter	Sm1	1537803675	17350.2	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	

Das Interpolationsverfahren STAMPED-LEFT ist der Default. Wird kein Interpolationsverfahren bei einer Auflösungsänderung angegeben, so wird mit diesem Interpolationsverfahren interpoliert.

Die Anfrage

```
SELECT * FROM SmartMeter [1537803000 : 5 : 1537803900]
```

liefert also das gleiche Ergebnis wie die oben stehende Anfrage.

5.9.2 Auflösungsveränderung mit Aggregation

Wird eine Zeitreihe mit einer neuen Auflösung abgefragt, so wird eine Aggregation durchgeführt, wenn die Auflösung der Datengrundlage höher ist, als die abgefragte Auflösung. Die Auflösung wird wie gehabt in der Intervallmitte angegeben und das Aggregationsverfahren hinter dem Intervall mit dem Schlüsselwort AGGREGATE BY gewählt. Die Liste möglicher Aggregationsverfahren und ihre Auswirkung ist Abschnitt 3.14 zu entnehmen.

```
SELECT * FROM SmartMeter [1537803000 : 1 : 1537805700] AGGREGATE BY AVG
```

Die Rückgabe listet alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche im genannten Zeitraum beziehungsweise auf den Grenzen liegen und aggregiert mittels des Verfahrens AVG.

Dieses berechnet den (nicht zeitgewichteten) Mittelwert, der ausgewählten, vorkommenden Zeitreiheneinträge:

timestamp	value
1537803000	17350.2
1537803900	17351.3
1537804125	17352.5
1537805700	17358.7

(17350.2 + 17351.3 + 17352.5 + 17358.7) / 4 = 17353.175

Und gibt diesen dementsprechend auf der Intervallmitte wieder:

type	mrid	timestamp	value	unit	multiplier	Category
SmartMeter	Sm1	1537804350	17353.175	Wh	K	

Das Interpolationsverfahren STAMPED-LEFT ist der Default. Wird kein Interpolationsverfahren bei einer Auflösungsänderung angegeben, so wird mit diesem Interpolationsverfahren interpoliert.

Die Anfrage

```
SELECT * FROM SmartMeter [1537803000 : 1 : 1537805700]
```

liefert also das gleiche Ergebnis wie die obige Anfrage.

5.9.3 Auflösungsveränderung mit Rasterung

Bei der Abfrage der Zeitreihen ist bei einer Auflösungsänderung auch eine Verschiebung des Rasters möglich. Die Auflösung wird wie gehabt in der Intervallmitte angegeben und das Rasterverfahren hinter dem Intervall mit dem Schlüsselwort RASTER BY gewählt. Die Liste möglicher Rasterverfahren und ihre Auswirkung ist Abschnitt 3.15 zu entnehmen.

```
SELECT * FROM SmartMeter [1537802750 : 3 : 1537805450) RASTER BY EVEN
```

Die Rückgabe berücksichtigt alle Daten aller Attribute von Zeitreiheneinträgen vom Typ SmartMeter, welche im genannten Zeitraum beziehungsweise auf den Grenzen und auf „geraden Uhrzeiten“ liegen (Rasterverfahren EVEN). In diesem Beispiel im halboffen gewählten Intervall sind die Randzeiten 45 Minuten voneinander entferrnt. Die 3 zwischenliegenden Werte sind deshalb 15 Minuten auseinander.

Die Randzeiten Timestamps entsprechen folgenden Uhrzeiten in ISO 8601:

- 1537802750: 2018-09-24T15:25:50+00:00 (im Folgenden „15:25:50“)
- 1537805600: 2018-09-24T16:10:50+00:00 (im Folgenden „16:10:50“)

3 „gerade“ Uhrzeiten im Zeitintervall sind:

- 2018-09-24T15:30:00+00:00 bzw. Zeitstempel 1537803000 (im Folgenden „15:30“)
- 2018-09-24T15:45:00+00:00 bzw. Zeitstempel 1537803900 (im Folgenden „15:45“)
- 2018-09-24T16:00:00+00:00 bzw. Zeitstempel 1537804800 (im Folgenden „16:00“)

Die Folgende Liste stellt die im gewählten Zeitraum vorhandenen Werte in der Zeitreihe dar:

timestamp	value	Zeit
1537803000	17350.2	15:30
1537803900	17351.3	15:45
1537804125	17352.5	15:48:45

Die Zeitreiheneinträge für die ersten beiden Zeitstempel („15:30“ und „15:45“) sind direkt vorhanden und in den Zwischenzeiten keine weiteren Einträge vorhanden, so dass nicht zu Aggregieren ist. Der Zeitreiheneintrag für den letzten Zeitstempel muss berechnet werden. Im Zeitraum zwischen 15:45 sowie der Grenze „16:10:50“ gibt es einen weitere verfügbaren Zeitreiheneintrag zum Zeitpunkt „15:48:45“. Eine Interpolation entfällt. Eine Aggregatgionsart ist nicht angegeben. Die Default-Aggregation AVG ergibt für den Zeitpunkt „16:00“ den Wert 17352.5 (als Mittelwert aller im Zeitraum vorkommenden Werte – also in diesem Fall der einzige Wert von 15:48:45).

Die Rückgabe für oben stehende Anfrage ist dementsprechend:

type	mrid	Timestamp	value	unit	multiplier	Category
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm1	1537804800	17352.5	Wh	K	

Die Default-Rasterfuntiktion ist „STRICT“ und verwendet zur Rasterung exakt die im Intervall angegebenen Randzeiten.

5.10 *QueryConditions in der SELECT Query*

5.10.1 WHERE

Mittels WHERE Klausel können Werte in Zeitreihen auf Bedingungen überprüft werden. Wird die Bedingung erfüllt, so wird der zugrundeliegende Zeitreiheneintrag weiterverarbeitet. Die Verwendung der WHERE-Klausel zur Einschränkung der zur Auswertung und Ausgabe zu verarbeitenden Zeitreiheneinträge ist nicht zwingend nötig. Es wird jedoch dringend empfohlen, wenn eine Zeiteinschränkung des Suchraumes nicht vorgenommen wurde. Ohne eine dieser Angaben, kann die Bearbeitung der Daten unter Umständen sehr langwierig werden, da erhebliche Daten zu verarbeiten und auszugeben seien können.

Das folgende Beispiel zeigt eine Möglichkeit des WHERE. Weitere Möglichkeiten sind in der W3School für SQL (Refsnes Data, 2018) einzusehen. Als Datengrundlage wird hier die Tabelle 1: SmartMeter Zeitreihe verwendet.

Im Beispiel sollen für eine Liste von Smartmeter-IDs zurückgegeben werden, ob zum aktuellen Zeitpunkt („NOW“ = 1537805710) Ersatzwerte vorliegen.

Die Anfrage lautet:

```
SELECT mrid, category FROM SmartMeter NEARESTBEFORE NOW WHERE SmartMeter.mrid IN
(„Sm1“, „Sm2“)
```

Die Rückgabe erscheint wie folgt:

mrid	category
Sm1	
Sm2	SubstitutedValue

Für Smartmeter Sm1 liegt kein Ersatzwert vor. Der Wert der vorliegt ist also ein gemessener Wert. Der für Sm2 abgelegte Wert ist ein Ersatzwert.

5.10.2 ORDER

Mittels ORDER Klausel können Einträge in Zeitreihen sortiert werden. Wird die ORDER-Klausel nicht explizit verwendet, ist der Default, dass das Ergebnis nach Zeitstempel aufsteigend sortiert ausgegeben wird. Als Datengrundlage wird hier die Tabelle 1: SmartMeter Zeitreihe verwendet.

Im Beispiel sollen die Liste der SmartMeter-Zeitreiheneinträge nicht chronologisch aufsteigend, sondern nach mrid-Bezeichner aufsteigend, jedoch nach Zeitstempel absteigend sortiert ausgegeben werden.

Die Anfrage lautet:

```
SELECT * FROM SmartMeter ORDER BY mrid ASC, timestamp DESC
```

Die Rückgabe erscheint wie folgt:

type	mrid	timestamp	value	unit	Multiplier	category
SmartMeter	Sm1	1537805700	17358.7	Wh	K	
SmartMeter	Sm1	1537804800	17355.0	Wh	K	
SmartMeter	Sm1	1537803900	17351.3	Wh	K	
SmartMeter	Sm1	1537803000	17350.2	Wh	K	
SmartMeter	Sm2	1537805700	2315.1	Wh	K	SubstitutedValue
SmartMeter	Sm2	1537804800	2312.2	Wh	K	
SmartMeter	Sm2	1537803900	2306.8	Wh	K	
SmartMeter	Sm2	1537803000	2301.4	Wh	K	

Die Ausgabe erfolgt sortiert in der Reihenfolge der angegebenen Sortierungen zunächst aufsteigend nach mrid und dann absteigend nach timestamp.

5.10.3 GROUP BY, HAVING, UNION und UNION ALL

Weitere Möglichkeiten für *QueryConditions* sind GROUP BY, HAVING, UNION oder UNION ALL. Beispiele dazu sind in der W3School für SQL (Refsnes Data, 2018) einzusehen.

6 Zusammenfassung

Das Projekt NetzDatenStrom hat den Bedarf an einer einheitlichen Anfragesprache für Zeitreihen offensichtlich gemacht, damit Zeitreihenarchive untereinander ausgetauscht werden können, aber auch neue Arten von Anfragen an Archive erfolgen können, ohne eine neue Implementierung für jede dieser Anfragen für das Archiv vornehmen zu müssen. So können anfragende Module und Archiv voneinander entkoppelt entwickelt werden. Die Anfragesprache wurde an SQL angelehnt definiert, um eine hohe Wiederverwendbarkeit hinsichtlich existierender SQL-Mechanismen sicherzustellen und eine Einstiegshürde bei möglichst vielen Anwendern gering zu halten. Aufgrund des Bekanntheitsgerades von SQL ist von einer schnellen Verwendbarkeit der Anfragesprache auszugehen. Im Wesentlichen wurden neue und einfache Möglichkeiten für die Abfrage von Zeiten und Zeiträumen, sowie der Interpolation, Aggregation und Rasterung von Zeitreihendaten beschrieben und des Weiteren neue Prefix- und Infix-Funktionen für Zeitreihendaten ermöglicht. Die Anfragesprache setzt keine relationalen Datenbanken voraus. Je nach unterliegender Technologie wird die Implementierung jedoch unterschiedlich komplex sein. Die einheitliche Anfragesprache stellt aber sicher, dass auf unterschiedlichen Technologien basierende Zeitreihenarchive austauschbar sind und gleich angesprochen werden können. Die Zeitreihenanfragesprache wird im Projekt NDS in Teilen umgesetzt, ist aber für die allgemeine Verwendung definiert.

7 Literaturverzeichnis

ISO. (1988). *ISO 8601 : 1988 (E): Data elements and interchange formats - Information interchange - Representation of dates and times.* International Standard: ISO.

ISO/IEC. (1996). *ISO/IEC 14977: 1996(E): Information technology - Syntactic metalanguage - Extendend BNF (1st Edition).* International Standard: ISO/IEC.

ISO/IEC. (2016). *ISO/IEC 9075 - SQL.* International Standard: ISO/IEC.

openKONSEQUENZ. (21. 09 2018). *openKONSEQUENZ*. Abgerufen am 21. 09 2018 von openKONSEQUENZ: https://www.openkonsequenz.de/

Refsnes Data. (21. 09 2018). *W3School SQL Tutorial*. Abgerufen am 21. 09 2018 von W3School: https://www.w3schools.com/sql/default.asp

Shafranovich, Y. (10 2005). *RFC 4180: Common Format and MIME Type for Comma-Seperated Values (CSF) Files*. (N. W. Group, Herausgeber) Abgerufen am 21. 09 2018 von https://tools.ietf.org/html/rfc4180

The Open Group. (21. 09 2018). *Base Specification Issue 7, 2018 edition, IEEE Std 1003.1-2017 (Revision of IEEE Std 1003.1-2008) - Chapter A4.16*. Abgerufen am 21. 09 2018 von The Open Group: http://pubs.opengroup.org/onlinepubs/9699919799/xrat/V4_xbd_chap04.html#tag_21_04_16

The OpenTSDB Authors. (21. 09 2018). *OpenTSDB - The Scalable Time Series Database*. Abgerufen am 21. 09 2018 von OpenTSDB: http://opentsdb.net/

8 Anhang

8.1 *EBNF Notation der Abweichungen der Anfragesprachensyntax zu SQL*

Im Folgenden werden die Teile der Anfragesprachensyntax, die von SQL abweichen, in erweiterter Backus Naur Form (eBNF) (ISO/IEC, 1996) dargestellt. Kommentare zu Schlüsselwörtern sind durch ein angehängtes „Comment“ im Schlüsselwort markiert. Die Kommentare dienen dem einfacheren Verständnis, sind aber bei einer Umsetzung auszulassen. Eine Umsetung bis in letzte Details der Darstellung von Namen und Zahlen wird aufgrund der besseren Lesbarkeit weggelassen und entsprechen dem Aufbau in SQL. An solchen Stellen findet sich ein Kommentar. Diese Stellen sind in der Umsetzung zu konkretisieren beziehungsweise von der unterliegenden SQL zu übernehmen.

```
SelectQuery ::= "SELECT" ("DISTINCT")? AttributeFunctionList "FROM" TypesList
TimeRestriction? QueryConditions? ("LIMIT" Number)?

AttributeFunctionList ::= "*" | AttributeFunction  ("AS" AttributeAliasName)?
(ListSeperator AttributeFunction  ("AS" AttributeAliasName)?)*
AttributeFunction ::= Attribute | Function
Attribute ::= AttributeName
AttributeName ::= ((Type | TypeAlias) "." Name) | Name
AttributeAliasName ::= Name

Function ::= PrefixFunction | InfixFunction
PrefixFunction ::= PrefixFunctionName ((Attribute | InfixFunction) | ( "(" (Value |
Attribute | InfixFunction |  PrefixFunction) (ListSeperator (Value | Attribute |
InfixFunction | PrefixFunction))*  ")"))
PrefixFunctionName ::= Name
InfixFunction ::= (Value | Attribute | PrefixFunction | "(" InfixFunction ")")
InfixFunctionName (Value | Attribute | PrefixFunction | "(" InfixFunction ")")
InfixFunctionName ::= Name


TypesList ::= Type ("AS" TypeAlias)?(ListSeperator Type ("AS" TypeAlias)?)* |
JoinCondition
Type ::= TypeName
TypeName ::= Name
TypeAlias ::= TypeAliasName
TypeAliasName ::= Name

JOINCondition ::= ("SQL Statements" JOIN)?
JOINConditionComment ::= "HierVereinfachtDargestellt"


TimeRestriction ::= TimeDirection? Time | TimeIntervalDefinition

TimeDirection ::= TimeDirectionName
TimeDirectionName ::= Name
Time ::= "NOW" | TimestampName | ISO8601ValueName
TimestampName ::= "Timestamp"
TimestampNameComment ::= "HierVereinfachtDargestellt"
ISO8601ValueName ::= "ISO(" "ISO8601-TIME" ")"
ISO8601ValueNameComment ::= "HierVereinfachtDargestellt"

TimeIntervalDefinition ::= ("(" | "[") StartTime? (TimeSeperator Resolution)?
TimeSeperator EndTime? (")" | "]") IntervalOptionals
StartTime ::= Time
EndTime ::= Time
```

```
Resolution ::= ISO8601Duration | Number
ISO8601Duration ::= "ISO8601-Duration"
ISO8601DurationComment ::= "HierVereinfachtDargestellt-LautWikipedia
[Startdatum]P[JY][MM][WW][TD][T[hH][mM][s[.f]S]]"
IntervalOptionals ::= ("INTERPOLATE BY" InterpolationFunction ( "("
InterpolationFunctionParameter (ListSeperator InterpolationFunctionParameter)* ")")?)?
("AGGREGATE BY" AggregationFunction ( "(" AggregationFunctionParameter (ListSeperator
AggregationFunctionParameter)* ")")?)? ("RASTER BY" RasterFunction ("("
RasterFunctionParameter (ListSeperator RasterFunctionParameter)* ")")?)?
InterpolationFunction ::= InterpolationFunctionName
InterpolationFunctionName ::= Name
AggregationFunction ::= AggregationFunctionName
AggregationFunctionName ::= Name
RasterFunction ::= RasterFunctionName
RasterFunctionName ::= Name
InterpolationFunctionParameter::= Value
AggregationFunctionParameter::= Value
RasterFunctionParameter::= Value


QueryConditions ::= ("SQLStatements" WHERE GROUPBY HAVING ORDER UNION)?
QueryConditionsComment ::= "HierVereinfachtDargestellt"
JOINS ::= See-SQL-Join-LeftJoin...
WHERE ::= See-SQL-WhereConditionsAndLogicalOperators
GROUPBY ::= See-SQL-Group-By
HAVING ::= See-SQL-HavingConditions
ORDER ::= See-SQL-Order
UNION ::= See-SQL-Union-And-Union-All


Number ::= "NUMBER"
NumberComment ::= "HierVereinfachtDargestellt"
Name ::= "NAME"
NameComment ::= "HierVereinfachtDargestellt"
Value ::= "VALUE"
ValueComment ::= "HierVereinfachtDargestellt"

ListSeperator ::= "," | ", "
TimeSeperator ::= ":" | ": " | " : "
```

8.2 *IntervallOptionals-Parameter*

Für die Interpolation, Aggregation und Rasterung sind bei einer Auflösungsänderung die Angaben von Parametern optional. Diese werden im Folgenden beschrieben. Die Parameter werden über die *IntervalOptionals* angegeben.

8.2.1 InterpolationFunctionParameter

Für die Interpolation sind bei einer Auflösungsänderung die Angaben von Parametern optional. Im Folgenden findet sich die Auflistung der Parameter für jede in Kapitel 3 genannte Interpolationsfunktion, **soweit Parameter nötig sind**. Werden diese Parameter nicht bei der Anfrage angegeben, so ist ein vorgegebener Default bei der Auswertung zu berücksichtigen. Dieser Default muss bei der Konzeptionierung der Interpolationsfunktion benannt und entsprechend umgesetzt werden. Die folgenden Tabellen enthalten je Interpolationsfunktion den Namen der Interpolationsfunktion, die Parameterposition, den Parameternamen, die Beschreibung des Parameters, dessen Wertebereich und Default-Wert, sowie den Typ des Wertes.

Interpolationsfunktion:		Name			
InterpolationFunctionName:		Name			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.2.2 AggregationFunctionParameter

Für die Aggregation sind bei einer Auflösungsänderung die Angaben von Parametern optional. Im Folgenden findet sich die Auflistung der Parameter für jede in Kapitel 3 genannte Aggregationsfunktion, **soweit Parameter nötig sind**. Werden diese Parameter nicht bei der Anfrage angegeben, so ist ein vorgegebener Default bei der Auswertung zu berücksichtigen. Dieser Default muss bei der Konzeptionierung der Aggregationsfunktion benannt und entsprechend umgesetzt werden. Die folgenden Tabellen enthalten je Aggregationsfunktion den Namen der Aggregationsfunktion, die Parameterposition, den Parameternamen, die Beschreibung des Parameters, dessen Wertebereich und Default-Wert, sowie den Typ des Wertes.

Aggregationssfunktion:		Name			
AggregationFunctionName:		Name			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.2.3 RasterFunctionParameter

Für die Rasterung sind bei einer Auflösungsänderung die Angaben von Parametern optional. Im Folgenden findet sich die Auflistung der Parameter für jede in Kapitel 3 genannte Rasterfunktion, **soweit Parameter nötig sind**. Werden diese Parameter nicht bei der Anfrage angegeben, so ist ein vorgegebener Default bei der Auswertung zu berücksichtigen. Dieser Default muss bei der Konzeptionierung der Rasterfunktion benannt und entsprechend umgesetzt werden. Die folgenden Tabellen enthalten je Rasterfunktion den Namen der Rasterfunktion, die Parameterposition, den Parameternamen, die Beschreibung des Parameters, dessen Wertebereich und Default-Wert, sowie den Typ des Wertes.

Rasterfunktion:		Name			
RasterFunctionName:		Name			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.2.4 Vorlage *IntervallOptionals*-Parametertabelle

Vorlage für die Parametertabelle:

***XYZ*-funktion:**		Umgangssprachliche Bezeichnung			
XYZ-functionName:		Bezeichnung in der Anfragesprache			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.3 *Dokumentation der Parameter mehrstelliger Prefix-Funktionen*

Die Parameter mehrstelliger Prefix-Funktionen werden in den folgendenen Unterkapiteln aufgelistet. Die folgenden Tabellen enthalten je Prefix-Funktion den Namen der Prefix-Funktion, die Parameterposition, den Parameternamen, die Beschreibung des Parameters, dessen Wertebereich und Default-Wert, sowie den Typ des Wertes. Je mehrstelliger Prefix-Funktion ist ein Unterabschnitt ab Abschnitt 8.3.2 mit einer Tabelle gemäß der Vorlage aus Abschnitt 8.3.1 aufzuführen. Weitere Erläuterungen zu der jeweiligen Prefix-Funktion können textuell neben den Tabellen erfasst werden.

8.3.1 Vorlage *PrefixFunction*-Parametertabelle

Vorlage für die Parametertabelle:

***Prefix-Funktion*:**		Umgangssprachliche Bezeichnung			
PrefixFunctionName:		Bezeichnung in der Anfragesprache			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.3.2 Prefix-Funktion „Name“

***Prefix-Funktion*:**		Name			
PrefixFunctionName:		Name			
Nr.	**Parametername**	**Beschreibung**	**Wertebereich**	**Defaultwert**	**Typ**

8.4 *In NetzDatenStrom benötigte Anfragesprachen-Elemente*

Für die Umsetzung der Anfragen der Visualisierungskomponenten in NetzDatenStrom sind die in der folgenden Tabelle beschriebenen Anfragesprachen-Operatoren nötig. Dazu wurden die Pseudo-Code-Anfragen auf die Anfragesprache gemappt. Anschließend wurden die nötigen Anfragesprachenelemente je Anfrage aufgelistet und dann in einer übergeordneten Liste zusammengetragen.

Kategorie	**Benötigtes Anfragesprachenelement**	**NDS-spezifischer Bedarf**
Auswahl	SELECT AttributeName	Auswahl eines Attributss mehrerer Zeitzeihen
Auswahl	SELECT DISTINCT AttributeName	Auswahl eines Attributs mehrerer Zeitreihen bei Rückgabe nur eines Wertes je gleichem Ergebnisses
Auswahl	SELECT AttributeList	Auswahl einer Liste von Attributen (ohne mathematische Funktionen) einer oder mehrerer Zeitreihen
Auswahl	SELECT *	Auswahl aller Attribute eines Types mehrerer Zeitreihen
Auswahl	FROM Typ	Festlegung des Types der Daten, auf die die Anfrage laufen soll
Auswahl	FROM TypListe	Festlegung auf einer Liste von Typen der Taten, auf die die Anfrage laufen soll
Zeit	NEARESTBEFORE	Auswahl des Wertes des nächsten vorherigen, oder gleichen Zeitstempels
Zeit	Zeitintervall (offen, geschlossen, halboffen)	Auswahl über ein Zeitintervall
Zeit	Timestamp, NOW, ISO8601-Zeitangabe	Angabe des Zeitpunkts bzw. Zeitintervallgrenzen
Zeit	(+-Unendlich)	Auswahl der Werte einer ganzen Zeitreihe bzw. von Beginn bis zu einem Zeitpunkt oder von einem Zeitpunkt bis zum Ende. Hier ist es sinnvoll, dies auf einzelne oder wenige Zeitreihen zu beschränken, da ansonsten insb. langfristig riesige Ausgaben zu erzeugen sind.
Zeit	Zeitintervall mit Auflösung (number, ISO 8601 Period)	Auflösungsänderung anhand einer neuen Anzahl von Stützstellen bzw. Zeitabständen, welche gemäß der ISO 8601 Period Definition angegeben werden.
Zeit	INTERPOLATE BY STAMPED-LEFT	Interpolation von Zeitreihenwerten im Intervall anhand Links gestempelter Werte auf die durch die Auflösung angegebene Anzahl von Stützstellen
Zeit	(AGGREGATE BY AVG)	Nicht explizit gefordert ist die Aggregation bei Auflösungsänderungen von Intervallen. Diese ist aber nötig für zu hoch aufgelöste Daten bei einer Neu-Rasterung des Intervalls. Hier muss der Default (AVG) umgesetzt werden, der aufgerufen wird, wenn das optionale AGGREGATE BY nicht im Anfragestring genant

		wird. Es wird also der Mittelwert aller im Zeitraum vorhandenen Werte unabhängig ihrer Gültigkeitsdauer gebildet.
Zeit	(RASTER BY STRICT)	Nicht explizit gefordert ist die Umrasterung bei Auflösungsänderungen von Intervallen. Diese ist aber nötig für Daten bei einer Neu-Rasterung des Intervalls. Hier muss der Default (STRICT) umgesetzt werden, der aufgerufen wird, wenn das optionale RASTER BY nicht im Anfragestring genant wird. Es werden also exakt die angegebenen Intervallgrenzen verwendet.
Bedingung	WHERE	Einleitung einer Bedingung
Bedingung	Attributwert = Wert	Vergleich auf Gleichheit eines Attributinhaltes mit einem festen Wert vom Typ Zeichenkette (mrid)
Bedingung	Attributwert < Wert	Vergleich auf Kleiner eines Attributinhaltes mit einem festen Wert vom Typ Zahl (plausibilityValue)
Bedingung	Attributwert IN Liste	Vergleich auf Anwesenheit eines Attributwertes in einer Liste fester Werte vom Typ Zeichenkette (mrid)
Bedingung	Attributwert LIKE Wert	Vergleich auf Übereinstimmung eines Attributwertes mit einem festen Wert vom Typ String (readingQualityTypeCategory)
Bedingung	Bedingung AND Bedingung	Und-Verknüpfung zweier vorgenannter Bedingungen (sowohl IN und LIKE, als auch IN und <)
Einschränkung	LIMIT 1	Begrenzung der Ausgabe auf eine einzelnen Wert

Die benötigten Anfragen sind:

1. **Aggregierter Subnetz-Status zu einem Zeitpunkt TS:**

   SELECT category FROM SmartMeter NEARESTBEFORE <TS> WHERE mrid IN (<*String*-Liste>) AND category LIKE *String* LIMIT 1

   Gibt eine leere Menge zurück, wenn kein Ersatzwert für eins der Smart Meter zu dem Zeitpunkt vorliegt, ansonsten eine einelementige Menge.

2. **Subnetz-Status zu einem Zeitpukt TS:**

   SELECT mrid, category FROM SmartMeter NEARESTBEFORE <TS> WHERE mrid IN (<*String*-Liste>)

   Gibt eine ungeordnete Menge von ID-Qualitätskategorie-Paaren zurück.
   Allgemein ist der default bei der Ordnung der Timestamp. Der ist aber nicht in der Ergebnismenge dieser Anfrage enthalten.

3. **Smartmeter Werte zwischen zwei Zeitpunkten TS1 und TS2 mit angegebener Auflösung Resolution**

   SELECT timestamp, value FROM SmartMeter [<TS1>:<Resolution>:<TS2>] INTERPOLATE BY STAMPED-LEFT WHERE mrid = *String*

   Gibt eine nach Zeitstempeln geordnete Menge zurück. Jedes zurückzugebendes Element in dieser Menge besteht aus dem Interpolationszeitstempel und dem interpolierten Wert.

4. **Aggregierter Subnetz-Status zwischen zwei Zeitpunkten TS1 und TS2**

   SELECT category FROM SmartMeter [<TS1>:<TS2>] WHERE mrid IN (<*String*-Liste>) AND category LIKE *String* LIMIT 1

   Gibt eine leere Menge zurück, wenn kein Ersatzwert für eins der Smart Meter in dem Zeitintervall vorliegt, ansonsten eine einelementige Menge.

5. **Subnetz-Status zwischen zwei Zeitpunkten TS1 und TS2**

   SELECT DISTINCT mrid FROM SmartMeter [<TS1>:<TS2>] WHERE mrid IN (<*String*-Liste>) AND category LIKE *String*

   Gibt die Liste aller mrids zurück, für die ein Ersatzwert in dem gegebenen Zeitraum vorliegt.

6. **Aktuelle Plausibilitätsdaten einer Liste von Smart Metern**

   SELECT mrid, timestamp, unit, plausibilityvalue, source, plausibilitytimestamp FROM Plausibility NEARESTBEFORE NOW WHERE mrid IN (<*String*-Liste>) AND plausibilityvalue < *Number*

   Gibt eine nach Zeitstempeln geordnete Menge zurück. Jedes zurückgegebenes Element enthält die Smart Meter ID, den Zeitstempel, die Einheit und die drei Plausiblitätsattribute. Kann alternativ auch nach meter.mrid sortiert werden, da es ja mehrere Plausibilisierungen pro Smart Meter geben kann.

7. **Wetterdaten eines Ortes zwischen zwei Zeitpunkten TS1 und TS2**

   SELECT * FROM Weather [<TS1>:<TS2>] WHERE location = *String*

   Gibt eine nach Zeitstempeln geordnete Menge der (wie auch immer aussehenden Wetterdaten) zurück.

Typische KiBiD-Queries:

1. **Alle Daten aus einer Zeitreihe zwischen zwei Zeitstempeln**

   SELECT * FROM <type> [<TS1>:<TS2>] WHERE mrid = *String*

2. **Alle Daten einer Menge von Zeitreihen zwischen zwei Zeitstempeln aber als Tabelle (d.h. Zeitstempel,ZR1.Wert,ZR1.Status,ZR2.Wert,ZR2.Status,...)**

   SELECT * FROM <type> [<TS1>:<TS2>] WHERE mrid IN (<*String*-Liste>)

3. **Berechnungen zwischen Zeitreihen (ist im Prinzip die Query zuvor)**

SELECT t1.Current*t2.Voltage AS Power FROM <type> AS t1, <type> AS t2 [<TS1>:<TS2>] WHERE mrid IN (<*String*-Liste>) AND t1.mrid = t2.mrid AND t1.timestamp = t2.timestamp AND t1.type = „Current“ AND t2.type = „Voltage“

4. **Zugriff auf den letzten Datenpunkt einer Zeitreihe.**

   SELECT * FROM <type> NEARESTBEFORE NOW WHERE mrid = *String*

5. **Abfragen der Coverage einer Zeitreihe (also Zeitstempel des ersten Datenpunktes und des letzten Datenpunktes einer Zeitreihe)**

   SELECT MIN timestamp AS Startzeitpunkt, MAX timestamp AS Endzeitpunkt FROM <type> WHERE mrid = String

8.5 *Typ-Definition*

Der Typ der Zeitreihen, die als *Type* in der *TypesList* angegeben wird, legt fest, welche Daten zu erwarten sind. Dazu setzt er sich aus zwei Komponenten zusammen:

1. Die Information, welche Daten vorhanden sind
2. Die Information, welche Daten eine Zeitreihe eindeutig charakterisieren.

Ähnlich wie in SQL *table* wird hier die Struktur der Daten angegeben. Diese Struktur findet sich jedoch abweichend zu SQL nicht zwingend in der Datenbank. Der Typ schränkt ausschließlich die Struktur ein, damit geeignete Anfragen formuliert und erwartungsgemäße Ergebnisse geliefert werden können.

Zusätzlich besitzt der Typ die Information, wie eine Zeitreihe (also eine Reihe von mit Zeitstempeln versehenen Werten) definiert ist, bzw. woran verschiedene Zeitreihen unterscheiden werden können.

Für SmartMeterPlausibility erfolgt eine Unterscheidung von Zeitreihen beispielsweise nach

- Meter.mrid,
- Reading.ReadingQualities.source oder/und
- ReadingType.unit

Für Weather erfolgt eine Unterscheidung von Zeitreihen beispielsweise nach

- EnvrionmentalMonitoringStation.mrid,
- Location.mainAdress.townDetail.name,
- EnvironmentalAnalog.unit oder/und
- EnvironmentalAnalog.kind

Das heißt:

1. jede Zeitreihe in SmartMeterPlausibility unterscheidet sich in einem der genannten Attribute.
2. Jeder Eintrag einer Zeitreihe in SmartMeterPlausibility hat die gleiche mrid, die gleiche source sowie die gleiche unit.
3. Gibt es in SmartMeterPlausibility Werte, die die gleiche mrid, die gleiche source sowie die gleiche unit haben, gehören sie zu derselben Zeitreihe.

Ein Format für die Typ-Definition wurde noch nicht festgelegt.