



Belastingdienst

RegelSpraak-specificatie

Datum:	24-1-2025
Versie	2.1.0

© 2024 Belastingdienst

Alle rechten voorbehouden.

Niets uit deze uitgave mag worden verveelvoudigd, opgeslagen in een geautomatiseerd gegevensbestand en/of openbaar gemaakt in enige vorm of op enige wijze, hetzij elektronisch, mechanisch, door fotokopieën, opnamen of op enige andere manier zonder voorafgaande schriftelijke toestemming van de auteur.

Inhoud

1	Versiebeheer.....	6
2	Inleiding	8
2.1	RegelSpraak - van idee tot realisatie	8
2.2	Opbouw document	10
2.3	Leeswijzer.....	11
2.3.1	Leeswijzer technische lezer	11
2.3.2	Leeswijzer functionele lezer	11
2.4	TOKA-casus.....	12
2.4.1	TOKA-belasting.....	12
2.4.2	Treinmiles.....	13
2.4.3	De TOKA-casus in RegelSpraak	13
3	GegevensSpraak - het objectmodel	14
3.1	Objecttypen	15
3.1.1	Bezield en onbezield.....	16
3.2	Eigenschappen van objecttypen	16
3.2.1	Attributen.....	17
3.3	Standaarddatatypes.....	17
3.3.1	Datatype Numeriek	18
3.3.2	Datatype Tekst	19
3.3.3	Datatype Boolean	19
3.3.4	Datatype Datum-tijd.....	20
3.3.5	Datatype Percentage	20
3.3.6	Voorbeelden bij standaarddatatypes.....	21
3.4	Domeinen	22
3.4.1	Domein op basis van een standaarddatatype	23
3.4.2	Domein op basis van een enumeratie	23
3.5	Kenmerken	24
3.6	Dimensies	26
3.7	Eenheden en eenheidssystemen	28
3.8	Tijdlijnen.....	30
3.9	Voorbeeld TOKA-casus: de objecttypen <i>Natuurlijk persoon</i> en <i>Vlucht</i> volledig gespecificeerd	31
3.10	Parameters	34
3.11	Feittypen en rollen	34
3.12	Dagsoort.....	36
4	De RegelSpraak-regels.....	37
4.1	Regel.....	37
4.2	Regelversie	37
4.3	Basispatroon.....	38
4.4	Voorbeeld: regels voor het attribuut leeftijd en de toekenning van het kenmerk <i>minderjarig</i>	39
5	Expressies	40
5.1	Tijdsafhankelijkheid	40
5.1.1	Tijdsafhankelijk versus niet-tijdsafhankelijk	40
5.1.2	Tijdsafhankelijkheid en eenheidssysteem Tijd	41
5.1.3	Specificatie van perioden	41
5.1.4	Resultaat tijdsafhankelijke expressies	43
5.2	Literal expressies	47
5.3	Rekendatum en Rekenjaar	48

5.4	Tekstreeks expressies	49
5.5	Onderwerpexpressies	49
5.5.1	Onderwerp	49
5.5.2	Selectie.....	50
5.5.3	Universeel onderwerp	50
5.5.4	Bezielde objecttypen	52
5.5.5	Onderwerpketen	53
5.6	Subselectie expressies	55
5.7	Concatenatie expressie	56
5.8	Aggregatie expressies	56
5.8.1	Telling van instanties: aantal	57
5.8.2	Sommatie: som van	58
5.8.3	Minimale/maximale waarde van	58
5.8.4	Eerste/laatste waarde: de eerste/laatste van	59
5.8.5	Aggregatie over dimensies	60
6	Rekenkundige expressies	61
6.1	Rekenen in RegelSprak.....	61
6.1.1	Rekenvolgorde	61
6.1.2	Getal-representatie	62
6.1.3	Afrondingen	63
6.1.4	Begrenzen van waarden	65
6.2	Optellen: plus	66
6.3	Aftrekken: min en verminderd met	67
6.4	Vermenigvuldigen: maal	68
6.5	Delen: gedeeld door en gedeeld door (ABS)	70
6.6	Worteltrekken: de wortel van	71
6.7	Machtsverheffen: tot de macht	72
6.8	Percentage bepalen: van	72
6.9	Absolute waarde van	73
6.10	Tijdsduur van datum-tijd tot datum-tijd	74
6.11	Datum plus/min tijdseenheid	75
6.12	Dag/maand/jaar uit	76
6.13	Eerste paasdag van	77
7	Tijdsafhankelijke expressies	78
7.1	Aggregeren in de tijd: totaal van.....	78
7.2	Tellen van dagen: het aantal dagen in ... dat	79
7.3	Naar verhouding van tijdsduur	80
7.3.1	Standaard omrekening tussen jaren en maanden	80
7.3.2	Omrekening met gebroken jaren of maanden – tijdsevenredig deel	80
8	Conditie en predicaten	84
8.1	Predicaten	84
8.1.1	Vergelijkingen.....	86
8.1.2	Lege waardes.....	87
8.1.3	De elfproef	87
8.1.4	Getalcontrole	88
8.1.5	Dagsoortcontrole	88
8.1.6	Uniciteit	88
8.1.7	Rolcheck.....	90
8.1.8	Kenmerkcheck	90
8.1.9	Regelpredicaat	90

8.2	Tijdsafhankelijke aanvulling – check op volledige periode	91
8.3	Enkele en samengestelde conditie	92
8.3.1	Enkele conditie	92
8.3.2	Samengestelde conditie	93
8.4	Tijdsafhankelijke conditie	95
8.4.1	Algemeen	95
8.4.2	Gedurende de tijd dat	96
8.4.3	vanaf / van...tot / van...tot en met / tot / tot en met	96
9	Resultaatdeel	98
9.1	Gelijkstelling	98
9.2	Kenmerktoekenning	99
9.3	ObjectCreatie	99
9.4	FeitCreatie	101
9.5	Consistentieregels	102
9.6	Initialisatie	103
9.7	Verdeling	104
9.7.1	Verdeling over alle ontvangers	104
9.7.2	Verdeling over groepen ontvangers in een volgorde	104
9.7.3	Gebruik van een maximale aanspraak	105
9.7.4	Gebruik van afronding	106
9.7.5	Gebruik van onverdeelde rest	106
9.7.6	Resultaten bij gebruik van lege of uitzonderlijke waarden	108
9.7.7	Syntax specificatie	108
9.8	Dagsoortdefinitie	108
9.9	Recurisie	109
10	Voorwaardendeel	110
10.1	Elementaire voorwaarde	110
10.2	Samengestelde voorwaarde	112
10.3	Tijdsafhankelijke voorwaarde	113
11	Variabelendeel	115
11.1	Rekenvolgorde variabelendeel	115
12	Beslistabellen	117
13	Formele specificatie van de RegelSpraak Syntax	121
13.1	Gebruikte notatie	121
13.1.1	Terminale symbolen	122
13.1.2	Niet-terminale symbolen	122
13.1.3	Definitie	123
13.1.4	Alternatief	123
13.1.5	Herhaling	123
13.1.6	Groepering	123
13.1.7	Optioneel	124
13.1.8	Controletekens	124
13.1.9	Overzicht gebruikte EBNF symbolen	124
13.2	Standaard syntax patronen	125
13.3	Objecten en parameters	125
13.3.1	Objecttypen	125
13.3.2	Attributen en kenmerken	126
13.3.3	Datatypen	126
13.3.4	Domeinen	126
13.3.5	Eenheden	126

13.3.6	Tijdslijnen	126
13.3.7	Dimensies.....	126
13.3.8	Parameters.....	126
13.3.9	Feittypen	127
13.3.10	Dagsoort.....	127
13.4	RegelSpraak.....	127
13.4.1	Onderwerpketen	127
13.4.2	RegelSpraak-regel.....	127
13.4.3	Resultaatdeel	127
13.4.4	Gelijkstelling.....	127
13.4.5	Kenmerktoekenning	127
13.4.6	ObjectCreatie	127
13.4.7	FeitCreatie.....	128
13.4.8	Consistentieregels	128
13.4.9	Initialisatie.....	128
13.4.10	Verdeling.....	128
13.4.11	Dagsoortdefinitie.....	128
13.4.12	Voorwaardendeel	128
13.4.13	Samengestelde voorwaarde	129
13.4.14	Elementaire voorwaarde	129
13.4.15	Berekening	132
13.4.16	Expressie	132
14	Lijst met figuren	134
15	Lijst met tabellen	135
16	Begrippenlijst.....	136
17	Index.....	137
	Bijlage 1: de TOKA-casus	139
	Bijlage 2: de EBNF alfabetten van CORBA(OMG), W3C en ISO/IEC 14977	143
	Bijlage 3: basisstructuur RegelSpraak in 3 EBNF varianten.....	144

1 Versiebeheer

Onderstaande tabel bevat het overzicht van gepubliceerde versies. In de omschrijving staan de wijzigingen ten opzichte van de vorige versie.

Versie	Status	Datum	Omschrijving	ALEF-versie
1.00	Definitief	01-05-2023	Initiële versie.	2022.3.0
1.0.1	Definitief	16-05-2023	Aanpassingen t.b.v. publicatie. Namen en informatie over concept-versies en mogelijke toekomstige aanpassingen verwijderd.	2022.4.0
1.1.0	Definitief	24-10-2023	<ul style="list-style-type: none"> Par. 3.3.1 - In specificatie numeriek datatype “reëel getal” gewijzigd in “getal”. Par. 3.10 - Aanduiding “binair” verwijderd bij feitttype. Ook verwijderd uit naamgeving in formele specificatie van syntax. Par. 5.1 - Enumeratiewaarde als aparte literal expressie toegevoegd in verband met presentatie met enkele aanhalingstekens. Par. 5.6 - Specificatie syntax Concatenatie expressie aangescherpt met gebruik “of” bij conditie met het predicaat “is gelijk aan”. Par. 6.1.4 - Begrenzingexpressie en expressies voor minimum en maximum begrenzing toegevoegd. Par. 7.1 - Tekstuele correctie vragende vorm elfproef in tabel met soorten predicaten. Par. 8.7 - Beschrijving resultaatdeel Verdeling aangescherpt en verduidelijkt. Par. 11 - Mogelijkheid van gebruik beslistabel voor kenmerktoekenning toegevoegd. 	2023.1.0
1.2.0	Definitief	11-04-2024	<ul style="list-style-type: none"> Specificatie van de expressie renteberekening verwijderd. Deze expressie wordt uitsluitend binnen de Belastingdienst gebruikt. Par. 5.7 Specificatie syntax van aggregatie gecorrigeerd. Par 6.7 Rekenkundige expressie voor machtsverheffen toegevoegd. Par 6.9 Rekenkundige expressie “absolute waarde van” toegevoegd. Par 6.10 Alternatief om gebruik te maken van “absolute tijdsduur van” toegevoegd Par. 6.11 Werking bij gebruik specifieke dagsoorten nader toegelicht. 	2024.1.0

Versie	Status	Datum	Omschrijving	ALEF-versie
2.0.0	Definitief	27-09-2024	<ul style="list-style-type: none"> Tijdsafhankelijk rekenen toegevoegd. Naast diverse kleine aanpassingen betreft dit de toevoeging van de volgende paragrafen: <ul style="list-style-type: none"> 3.8 Tijdlijnen 5.1 tijdsafhankelijkheid in expressies 7.1 / 7.2 / 7.3 Specifieke tijdsafhankelijke expressies 8.2 Check op volledige periode 8.4 Tijdsafhankelijke conditie 10.3 Tijdsafhankelijke voorwaarde 	2024.2.0
2.1.0	Definitief	25-1-2025	<ul style="list-style-type: none"> Par. 8.1.7 Mogelijkheid om "heeft" te gebruiken bij rolcheck toegevoegd. Par. 9.9 met beschrijving recursie toegevoegd. Hfst. 11 Mogelijkheid van lidwoord bij naam variabele toegevoegd. Diverse beschrijvingen tijdsafhankelijke berekeningen aangescherpt. 	2024.3.0

Tabel 1: historie specificatiedocument.

2 Inleiding

Dit document beschrijft **GegevensSpraak** en **RegelSpraak**. Beide zijn formele natuurlijke talen die ontwikkeld zijn binnen de Belastingdienst. Met GegevensSpraak leg je vast met welke **gegevens** gerekend wordt en welke gegevens het resultaat zijn. Met RegelSpraak is het mogelijk om **regels** te formuleren, al dan niet op basis van wet- en regelgeving, die voor iedereen die de Nederlandse taal beheerst leesbaar en begrijpelijk zijn. Kortom, GegevensSpraak specificeert de gegevens die gebruikt worden in RegelSpraak.

Doordat GegevensSpraak en RegelSpraak formele talen zijn, kunnen de regels met behulp van een generator, die de betreffende meta beheerst, worden omgezet in software die de regels uitvoert.

Voor de toepassing van GegevensSpraak en RegelSpraak is binnen de Belastingdienst de tool ALEF (Agile Law Execution Factory) ontwikkeld. De specificatie van GegevensSpraak en RegelSpraak in dit document is gebaseerd op de implementatie in ALEF. Zie hoofdstuk 1 voor de ALEF-versie behorend bij dit document.

Een korte schets van het ontstaan van RegelSpraak, met in zijn kielzog GegevensSpraak, is opgenomen in navolgende paragraaf.

2.1 RegelSpraak - van idee tot realisatie



Binnen de Belastingdienst leefde al heel lang het idee om regels eenduidig op te slaan en geautomatiseerd te kunnen vertalen. In 2009 begon dit concreet vorm te krijgen doordat Elian Baars (een beleidsmedewerker) en Frans Fokkenrood (een medewerker met een technische achtergrond) met elkaar in contact kwamen. Beiden waren erg enthousiast over dit idee en samen begonnen zij aan de ontwikkeling van Beheer van Regelspecificaties.

De eerste probeersels met betrekking tot beheer van regelspecificaties waren op basis van SBVR (**S**emantics **O**f **B**usiness **V**ocabulary **A**nd **B**usiness **R**ules) en RuleSpeak. Zij liepen hierin uiteindelijk vast omdat deze taal niet strikt genoeg bleek te zijn. De taal moest verder uitgebreid worden zodat de gewenste striktheid en precisie hierin aanwezig was. Zo werd eind 2009 RegelSpraak geboren. Irma Volkers (teamleidster) zag potentie in het idee en ondersteunde het initiatief met het aantrekken van extra expertise in de persoon van Gertrude Sangers – van Cappellen in 2010.

Gertrude Sangers – van Cappellen gaf bij het team het RegelSpraak document meer en meer vorm. Het document van RegelSpraak groeide en zo konden al snel meerdere (regel)patronen ondersteund en toegepast worden. Hierbij werd gebruik gemaakt van de applicatie RuleXpress (een tool om het vocabulaire, regels en metadata te beheren). Frans Fokkenrood bouwde hierop zijn eigen software met de tooling ANTLR (**A**Nother **T**ool for **L**anguage **R**ecognition) en o.a. Eclipse. Zo konden taalpatronen herkend, gecontroleerd en vertaald worden naar (computer)code.

In 2010 werden de regels voor Fiscale voorControle voor de InkomensHeffing omgezet naar RegelSpraak-regels. In deze periode werden de eerste resultaten van RegelSpraak zichtbaar en werden de eerste presentaties binnen en buiten de Belastingdienst gehouden. Dit was een eerste belangrijke stap en de reviews en op- en aanmerkingen waren zeer waardevol en bruikbaar voor de verdere ontwikkeling van RegelSpraak.

In 2011/2012 ontwikkelden Diederik Dulfer en Peter Straatsma een visie op het toegankelijk, beheersbaar en herbruikbaar maken van de kennis in wet- en regelgeving als basis voor specificaties voor geautomatiseerde ondersteuning van de uitvoering. Toepassing van RegelSpraak had daarin een centrale rol. Deze ontwikkeling kreeg onder de naam [Wendbare Wetsuitvoering](#) verder vorm.

In 2014 werd commitment verkregen bij het management om toepassing van RegelSpraak en de ontwikkelde aanpak concreet te gaan beproeven.

Begin 2015 startte de Proof of Concept Zorgverzekeringswet. In deze Proof of Concept werd de hele aanpak (van wet tot implementatie) van Wendbare Wetsuitvoering beproefd, met de beperking dat de regels aan moesten sluiten bij de bestaande implementatie. In deze Proof of Concept werd de eerste versie van ALEF (**A**gile **L**aw **E**xecution **F**actory) ontwikkeld onder leiding van Gert Veldhuizen van Zanten. Regels uit RuleXpress werden handmatig overgenomen in ALEF. In ALEF werden ze geëxecuteerd en gevalideerd en vervolgens met de codegenerator omgezet in Blaze-rules.

In fase 2 van de Proof of Concept Zorgverzekeringswet, in 2017, werd er meer focus gelegd op de wetsanalyse. De opzet van het project was geheel conform de beoogde werkwijze van wetgeving, naar regels, naar valideren, naar testen, naar programmatuur. Precies zoals de Wendbare Wetsuitvoering dit voor ogen had.

De Proof of Concept was succesvol en de ontwikkeling van de methode en tooling mocht worden doorgezet. ALEF werd uitgebreid met functionaliteit om regels op te stellen. De volgende stap was de inrichting van de rekenservice InkomensHeffing (IH) die op basis van deze eerste RegelSpraak-ideeën en -patronen volledig met ALEF werd ontwikkeld. Ook deze proef was succesvol. De rekenservice IH is sinds 2019 in productie.

Met de succesvolle ontwikkeling van deze zeer complexe service werd aangetoond dat een aanpak was ontwikkeld die binnen de Belastingdienst breed kon worden ingezet.

Tegelijkertijd was in 2018 ook gestart met het project “Uitnodiging tot het doen van aangifte”, ook wel Beschrijven en Attenderen genoemd. Dit betrof een wat “verouderd” systeem met query’s. Deze query’s zijn uit de code gehaald en omgezet in beschreven beleid. Dit beleid is vervolgens weer herschreven in RegelSpraak-regels. De samenwerking met de beleidsmedewerkers en medewerkers in de uitvoering was hier een verrijking van de algehele methode van Wendbare Wetsuitvoering.

In de jaren daarna is RegelSpraak verder ontwikkeld op basis van behoeften die naar voren kwamen bij het opstellen van regels voor een groot aantal domeinen binnen de Belastingdienst (o.a. InkomensHeffing, LoonHeffing, Inning, Toeslagen, Schenk- en Erfbelasting, Milieubelasting).

Op dit moment wordt gewerkt aan de toevoeging van taalpatronen voor tijdsafhankelijke berekeningen aan RegelSpraak. Taalpatronen waarmee ook al tijdens de Proof of Concept Zorgverzekeringswet is geëxperimenteerd. In de algemene aanpak van Wendbare Wetsuitvoering wordt daarbij gestreefd naar samenwerking in multidisciplinaire teams waarbij regels worden gespecificeerd conform wet- en regelgeving.

Het resultaat van de ontwikkeling van RegelSpraak samen met beleidsmedewerkers en medewerkers in de uitvoering wordt geïllustreerd met de volgende uitspraak van een betrokken beleidsmedewerker: *“Ik kan nu de regels lezen en interpreteren. Ik kan ze aanpassen, naar het beleid zoals men dat nu zou willen. Het is leesbaar, testbaar en vooral ook herbruikbaar. Wat nog belangrijker is, het is ook nog eens te herleiden naar de wet en het beleid. Zo is de cirkel van de Wendbare Wetsuitvoering pas echt rond!!”*.

2.2 Opbouw document

Dit document is incrementeel opgebouwd. Dat betekent concreet dat een nieuw te introduceren begrip (concept) zoveel mogelijk voortbouwt op zaken die eerder aan bod zijn gekomen. In dit document zal derhalve vanaf hoofdstuk 3 bij het introduceren van een nieuw begrip het eerste voorkomen ervan **vet** worden weergegeven. Hoofdstuk 16 bevat een begrippenlijst waarin begrippen die niet in de tekst zijn uitgelegd, zijn gedefinieerd.

Elk RegelSpraak concept wordt kort en bondig beschreven, vergezeld van een concreet RegelSpraak voorbeeld. Voor de voorbeelden in dit document wordt de TOKA (Treinen Op Korte Afstand) casus gebruikt, die als “Bijlage 1: de TOKA-casus” in zijn volledigheid is toegevoegd en hierna in paragraaf 2.4 nader wordt toegelicht.

In ALEF wordt een aantal kleuren toegepast om de verschillende RegelSpraak aspecten te onderscheiden. Deze kleuren gebruiken we in dit document ook ter verduidelijking en leesbaarheid in de RegelSpraak voorbeelden.

1. **Zwart vetgedrukt** wordt gebruikt voor RegelSpraak woorden in declaraties;
2. Zwart wordt gebruikt voor lidwoorden, types, en andere RegelSpraak uitdrukkingen;
3. **Paars** wordt gebruikt voor de namen van objecttypen;
4. **Groen** wordt gebruikt voor namen van attributen en voor waarden van enumeraties;
5. **Oranje** wordt gebruikt voor namen van kenmerken en voor namen en waarden van dimensies;
6. **Blauw** wordt gebruikt voor rollen en namen van parameters.

Als er in de tekst verwezen wordt naar termen uit GegevensSpraak, RegelSpraak of de voorbeelden, dan zal dit *cursief* worden weergegeven.

Tot slot is een formele specificatie van de RegelSpraak syntax opgenomen in dit document. Formele specificaties van de syntax zijn hierbij in eerste instantie verweven in de tekst. Deze onderdelen staan beschreven in een kader met een grijze achtergrond zoals hieronder afgebeeld. De syntaxspecificatie staat niet in directe relatie met de gebruikte voorbeelden in een paragraaf. Om een voorbeeld uit te werken met de syntaxspecificaties zijn alle in het document eerder beschreven syntaxspecificaties nodig. Het kan echter ook zo zijn dat er in een patroon een variabel deel uitgelegd staat wat nog niet aan bod is gekomen. Voor de volledigheid is/wordt die dan wel getoond.

Syntax voorbeeld:

```
<extra opmerking>

<voorbeeld 1> ::= "dit is een eerste voorbeeld"
<voorbeeld 2> ::= <bepaaldlidwoord> <naam object>
```

Niet alle syntaxdefinities staan overigens in de tekst verweven; voor een totaaloverzicht van de volledige RegelSpraak syntax wordt verwezen naar hoofdstuk 13. Voor een handige lijst van de

hierbij gebruikte EBNF symbolen wordt verwezen naar paragraaf 13.1.9.

Tot slot wordt opgemerkt bij de syntaxdefinities dat spaties tussen de variabele delen (aangeduid met vishaken < en >) niet expliciet opgenomen zijn middels “ ”, ter bevordering van de leesbaarheid van deze syntaxdefinities. Uiteraard zullen deze wel toegepast moeten worden bij het opstellen van RegelSpraak-regels. Bijvoorbeeld: we hanteren dus <bepaaldlidwoord> <naam object> in plaats van <bepaaldlidwoord> " " <naam object>.

2.3 Leeswijzer

In algemene zin is dit specificatiedocument bedoeld voor twee groepen lezers:

- een technische lezer die een implementatie wil maken om RegelSpraak-regels te specificeren of uit te voeren;
- een functionele lezer zoals een regelanalist die met name RegelSpraak-regels wil leren lezen en een geschreven RegelSpraak-regel wil (kunnen) begrijpen, maar ook in basis RegelSpraak-regels kan opstellen.

2.3.1 Leeswijzer technische lezer

Voor de technische lezer is het gehele document van belang:

- Hoofdstuk 3 beschrijft het objectmodel dat gebruikt wordt in een RegelSpraak-regel.
- Hoofdstuk 4 beschrijft de basisstructuur waaruit een RegelSpraak-regel bestaat.
- Hoofdstuk 5 behandelt diverse soorten expressies die in RegelSpraak-regels worden gebruikt waarna in hoofdstuk 6 de specifieke en veelgebruikte rekenkundige expressies separaat zijn beschreven.
- Hoofdstuk 7 beschrijft het gebruik van condities en predicaten in een RegelSpraak-regel.
- Hoofdstuk 9 beschrijft het resultaatdeel van een RegelSpraak-regel en hoe expressies gebruikt kunnen worden als actie van een dergelijke regel.
- Hoofdstuk 9.9 beschrijft het voorwaardendeel van een RegelSpraak-regel en hoe expressies in voorwaarden gebruikt (kunnen) worden.
- Hoofdstuk 11 beschrijft het variabelendeel en hoe dit gebruikt kan worden om regels overzichtelijker te maken.
- Hoofdstuk 12 beschrijft het gebruik van een beslistabel waarmee doorgaans meerdere RegelSpraak-regels in hun samenhang worden gerepresenteerd.
- Hoofdstuk 13 bevat een totaaloverzicht van de syntaxdefinities van RegelSpraak; wellicht handig deze als technische lezer als eerste door te nemen als ondersteuning bij het doornemen van de grijze syntaxblokken.
- Het document wordt tot slot afgesloten met diverse overzichtshoofdstukken met verwijslijsten met figuren en tabellen, een index, een begrippenlijst en enkele bijlagen ter ondersteuning van de specificaties.

2.3.2 Leeswijzer functionele lezer

Voor de functionele lezer, die RegelSpraak-regels wil leren lezen en een geschreven RegelSpraak-regel wil (kunnen) begrijpen, bevelen wij aan om in ieder geval de hoofdstukken 4, 9, 9.9 en 11 door te nemen waarbij de grijze blokken met syntaxspecificaties overgeslagen kunnen worden. Voor concepten die meer uitleg behoeven kunnen de hoofdstukken die overgeslagen zijn geraadpleegd worden. Ook kan via de index of de begrippenlijst achterin het document gezocht worden naar uitleg van enkele begrippen.

De functionele lezer die daarnaast ook RegelSpraak-regels wilt kunnen opstellen, adviseren we om daarnaast de hoofdstukken 5, 6 en 7 door te nemen, eventueel uitgebreid met hoofdstuk 12

over het gebruik van beslistabellen.

Uiteraard staat het de functionele lezer vrij om het gehele document door te nemen.

2.4 TOKA-casus

De TOKA-casus beschrijft de wet Treinen Op Korte Afstand. In Bijlage 1: de TOKA-casus is de volledige casustekst opgenomen, deze paragraaf bevat een toelichting hierop.

De wet Treinen Op Korte Afstand is een fictieve wet die geen enkele inhoudelijke relatie heeft met werkelijke belastingwetgeving, maar qua structuur wel aansluit bij de werkelijkheid.

Daarom is deze casus goed te gebruiken als rode draad in dit specificatiedocument. Hierbij wordt wel als kanttekening opgemerkt dat voorbeelden hieruit ter illustratie dienen bij de uitleg van een specifiek GegevensSpraak of RegelSpraak onderdeel.

De gedachte achter deze fictieve wet is dat, hoewel vliegen een snelle manier van transport is, het erg belastend is voor het milieu. Binnen Europa is de trein een goed alternatief voor korte vluchten. Om treingebruik te stimuleren, stelt de regering een nieuwe belasting in: de 'Treinen Op Korte Afstand' oftewel TOKA-belasting. De opbrengsten hiervan kunnen vervolgens worden gebruikt voor het verder verbeteren van het treinnetwerk.

De fictieve wet die deze belasting beschrijft, bestaat uit twee hoofdonderdelen. Het eerste hoofdonderdeel stelt dat of de passagiers van een vlucht, of de luchtvaartmaatschappij, onder bepaalde omstandigheden een belasting moet(en) betalen¹. Dit hoofdonderdeel beschrijft de regels waarmee het belastingbedrag kan worden vastgesteld. Het tweede hoofdonderdeel stelt dat onder de passagiers van een vlucht treinmiles worden verdeeld. Treinmiles zijn in de fictieve wet een betaalmiddel voor het aanschaffen van vervoersbewijzen in het openbaar vervoer. Dit hoofdonderdeel beschrijft de regels voor de verdeling.

Verder geeft de fictieve wet aan wanneer voldaan dient te worden aan de aangifte van de belasting en betaling hiervan.

Hieronder wordt kort beschreven welke concepten gebruikt worden in de TOKA-casus en hoe deze van invloed zijn op de TOKA-belasting (zie paragraaf 2.4.1) en treinmiles (zie paragraaf 2.4.2). Het doel hiervan is om bekendheid met de casus en concepten te creëren, zodat duidelijk is waar de voorbeelden die later in dit document gebruikt worden, vandaan komen. De volledige tekst van de TOKA-casus is toegevoegd in Bijlage 1: de TOKA-casus.

2.4.1 TOKA-belasting

Factoren die bepalen of een passagier een TOKA-belasting moet betalen, zijn:

1. vluchtafstand: indien de vluchtafstand groter is dan een vaste waarde, wordt geen belasting geheven;
2. bereikbaarheid van het eindpunt van de vlucht per trein: indien het eindpunt niet bereikbaar is per trein, wordt geen belasting geheven;
3. vluchttype: indien de vlucht een rondvlucht is, wordt geen belasting geheven van de passagiers.

De luchtvaartmaatschappij betaalt overigens alleen zelf een TOKA-belasting indien de vlucht een rondvlucht is.

¹ Dit betreft dus een voorbeeld van het fictieve deel van deze TOKA-casus: hierbij wordt namelijk uitgegaan van de combinatie van een vlucht (enkelvoud) en meerdere personen (meervoud) terwijl in de werkelijkheid in principe meerdere personen meerdere vluchten kunnen boeken. Voor de eenvoud en duidelijkheid qua uitleg van RegelSpraak is gekozen dit achterwege te laten.

Indien een passagier een TOKA-belasting moet betalen, wordt deze berekend aan de hand van de volgende drie onderdelen:

- a. belasting op basis van afstand;
Dit is afhankelijk van:
 1. vluchtafstand: hoe korter de vluchtafstand, des te groter het belastingbedrag;
 2. leeftijd: de tarieven die gebruikt worden zijn afhankelijk van de leeftijdscategorie waar de passagier bij hoort;
- b. vermeerdering belasting op basis van reisduur per trein;
Het belastingbedrag op basis van afstand kan vermeerderd worden afhankelijk van de reisduur per trein;
- c. duurzaamheidskorting;
Of een passagier hier recht op heeft, is afhankelijk van:
 1. de brandstofsamenstelling van de vlucht;
 2. de leeftijd van de passagier;
 3. de vluchtafstand.

Daarnaast is van belang hoe om te gaan met tariefswijzigingen van de TOKA-belasting waarbij de boekingsdatum van belang is, zie hiervoor verder Bijlage 1: de TOKA-casus.

2.4.2 Treinmiles

Voor iedere vlucht wordt een hoeveelheid treinmiles beschikbaar gesteld om te verdelen onder de passagiers. Deze treinmiles kunnen vervolgens worden gebruikt voor het aanschaffen van een vervoersbewijs voor het openbaar vervoer. Het aantal treinmiles dat beschikbaar is, bestaat uit een vaststaand deel en een variabel deel dat afhankelijk is van de hoeveelheid passagiers per vlucht. Treinmiles worden verdeeld aan de hand van leeftijd: hoe hoger de leeftijd, des te kleiner het aantal treinmiles dat de passagier krijgt. Bij gelijke leeftijd wordt een woonregio-factor toegewezen. Deze woonregio-factor is afhankelijk van de provincie waar de passagier woont.

2.4.3 De TOKA-casus in RegelSpraak

De concepten die voorkomen in de fictieve TOKA-casus zijn overigens niet voldoende om voorbeelden te geven van alle soorten regels die gebruikt kunnen worden in RegelSpraak. In de werkelijkheid komt het ook voor dat een wet niet toereikend is voor het afdekken van alle situaties en details die voorkomen in de praktijk. In dat geval kan uitvoeringsbeleid worden opgesteld.

Vergelijkbaar worden in de fictieve TOKA-casus ook additionele concepten onderkend die ervoor zorgen dat wel alle situaties en details kunnen worden afgedekt. Voor de fictieve TOKA-casus bijvoorbeeld zou dat kunnen zijn de uiterste boekingsdatum van een vlucht en het aantal dagen dat nodig is voor de verwerking van een boeking. De uiterste boekingsdatum voor een vlucht kan logischerwijs niet later zijn dan de vluchtdatum, maar ook moet rekening gehouden worden met de tijd die de luchtvaartmaatschappij nodig heeft voor het verwerken van de boeking. Hoewel deze concepten dus niet in de fictieve wettekst voorkomen, zijn ze wel relevant voor dit onderwerp en geeft het meer mogelijkheden voor het laten zien van de mogelijkheden die RegelSpraak biedt voor regels die met tijdsduur te maken hebben dan de oorspronkelijke TOKA-casus.

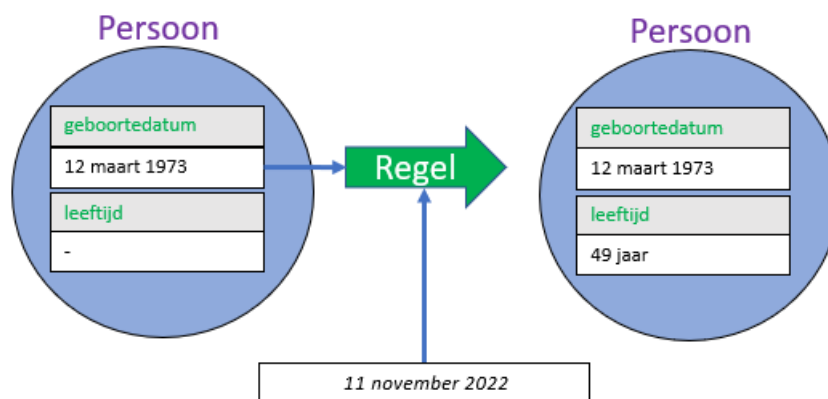
3 GegevensSpraak - het objectmodel

De meeste regels die in RegelSpraak zijn of worden gemaakt, worden toegepast op **objecten**. Objecten zijn representaties van dingen in de echte wereld. Veel wetgeving zal bijvoorbeeld gaan over personen. In RegelSpraak worden de werkelijke personen abstract gerepresenteerd door persoonsobjecten. Deze objecten hebben de eigenschappen die deze personen hebben en die relevant zijn voor de context (waaronder wetgeving) die gemodelleerd moet worden. Vervolgens kunnen er RegelSpraak-regels gemaakt worden die de eigenschappen van deze persoonsobjecten gebruiken of veranderen.

De objecten en hun eigenschappen worden gespecificeerd met de taal **GegevensSpraak**. Dit is weliswaar een andere taal dan RegelSpraak, maar we zullen het toch in dit hoofdstuk beschrijven aangezien RegelSpraak de objecten uit GegevensSpraak nodig heeft om te kunnen functioneren.

De waarden van eigenschappen kunnen afkomstig zijn van daadwerkelijke waarden in de echte wereld. De objecten voor personen kunnen bijvoorbeeld een eigenschap *geboortedatum* hebben. Voordat RegelSpraak-regels op de persoonsobjecten kunnen worden uitgevoerd zal deze eigenschap voor elk persoonsobject een waarde moeten krijgen die overeenkomt met de geboortedatum van een persoon in de echte wereld. Hoe dit precies in zijn werk gaat, is implementatie en valt daarmee buiten het bestek van dit document.

De waarden van eigenschappen kunnen daarnaast ook door RegelSpraak-regels bepaald worden. Als we bijvoorbeeld een object *Persoon* hebben gespecificeerd met de eigenschappen *geboortedatum* en *leeftijd*, en ervoor hebben gezorgd dat de eigenschap *geboortedatum* de juiste waarde heeft, dan kunnen we een RegelSpraak-regel maken die de eigenschap *leeftijd* van dit persoonsobject bepaalt op een bepaalde datum. Zie navolgende illustratie.



Figuur 1: een RegelSpraak-regel toegepast op een object *Persoon*. De regel bepaalt de eigenschap *leeftijd* waarbij de eigenschap *geboortedatum* wordt gebruikt. Daarnaast gebruikt de regel een datum waarop de leeftijd bepaald moet worden, in dit geval 11 november 2022.

Objecten in RegelSpraak worden niet opgeslagen, RegelSpraak en GegevensSpraak zijn namelijk geen databases maar specificatietalen waarmee objecten gespecificeerd kunnen worden. Als dezelfde regel wordt uitgevoerd met een andere datum waarvoor de *leeftijd* moet worden berekend, bijvoorbeeld 12 maart 2023, dan zal als resultaat van die regel de *leeftijd* de waarde 50 jaar krijgen. De eigenschap *geboortedatum* houdt dezelfde waarde aangezien deze in de echte wereld (ook) niet verandert.

3.1 Objecttypen

Objecten in RegelSpraak en GegevensSpraak zijn altijd **instanties** van bepaalde **objecttypen**. Een objecttype definieert met welke term een groep gelijksoortige objecten aangeduid wordt, oftewel hoe het objecttype genoemd wordt, en specificeert welke eigenschappen elk object van dit objecttype heeft. Als voorbeeld geldt het objecttype Persoon die in de inleiding van dit hoofdstuk is benoemd.

In een RegelSpraak-regel wordt dit objecttype vervolgens gebruikt om aan te geven op welke objecten de regel toegepast moet worden. In een RegelSpraak-regel kan geen specifieke instantie worden gebruikt. In plaats daarvan wordt een verwijzing naar het objecttype gebruikt. De regel wordt dan opnieuw uitgevoerd voor elke instantie van dat objecttype. Hierbij kan direct naar het objecttype verwezen worden met behulp van de eigenschappen die het objecttype specificeert en de rollen die het objecttype kan spelen (zie paragraaf 3.11).

Bij ons voorbeeld uit Figuur 1 zouden we een objecttype Persoon voor de persoonsobjecten kunnen specificeren dat de eigenschappen geboortedatum en leeftijd beschrijft. We kunnen daarmee een RegelSpraak-regel schrijven om de leeftijd te bepalen op basis van de geboortedatum waarin we gebruik maken van dat objecttype. Bij het uitvoeren van RegelSpraak worden instanties van dit objecttype beschikbaar aan de hand van de invoergegevens. Bij het uitvoeren van RegelSpraak zal voor elk van de instanties de regel worden uitgevoerd: met andere woorden bij elke instantie wordt de leeftijd berekend op basis van de geboortedatum. Een RegelSpraak-regel is wat dat betreft in basis een generieke uitspraak die wordt toegepast op alle bestaande en relevante instanties van de genoemde objecttypen.

Een **objecttypedefinitie** begint met het woord *Objecttype*, gevolgd door een **naamwoord**. Het RegelSpraak voorbeeld uit de TOKA casus dat wordt gehanteerd bij het uitleggen van de opbouw van een naamwoord is het objecttype *Natuurlijk persoon* met de volgende RegelSpraak definitie:

Objecttype de *Natuurlijk persoon* (mv: Natuurlijke personen)

- 1) Een naamwoord bevat altijd een zelfstandig naamwoord of naamwoorden, zeg maar de naam. In bovenstaand voorbeeld is de naam *Natuurlijk persoon*.
- 2) Voor de naam kan optioneel een bepaald lidwoord gespecificeerd worden. In bovenstaand voorbeeld is het lidwoord *de* gebruikt.
- 3) Een naamwoord heeft daarnaast een meervoudsvorm. Deze wordt achter de naam gespecificeerd door de tekst van deze meervoudsvorm tussen haakjes te zetten en *mv:* ervoor te zetten². In bovenstaand voorbeeld is de meervoudsvorm *Natuurlijke personen*.

Syntax naamwoord specificatie:

```
<naamwoord> ::= [ <bepaaldlidwoord> ] <naam> " (mv:" <meervoudsvorm>)" "
<bepaaldlidwoord> ::= "de" | "het"
<naam> ::= <karakterreeks>
<meervoudsvorm> ::= <karakterreeks>
```

Het naamwoord wordt niet alleen gebruikt om een bepaald objecttype te specificeren, maar ook om naar instanties van het bepaalde objecttype te verwijzen. Een objecttype om een persoon te representeren is bijvoorbeeld hierboven weergegeven. Als we vervolgens in een regel gebruik willen maken van een specifieke instantie of meerdere specifieke instanties van

² ALEF specifiek: mocht er geen meervoudsvorm gespecificeerd zijn, dan zal ALEF deze proberen af te leiden van de naam. Dit staat echter los van RegelSpraak.

dit objecttype, dan gebruiken we *de Natuurlijk persoon* of *de Natuurlijke personen*.

3.1.1 Bezielt en onbezielt

Bij een objecttype kan verder aangegeven worden of het objecttype **bezielt** is. Hiermee wordt aangegeven of het betreffende objecttype een levend wezen is, oftewel een objecttype met een ziel. Naar bezielde objecttypen kan vervolgens verwezen worden met een verwijzend voornaamwoord 'hij' en een bezittelijk voornaamwoord 'zijn'. Hoe dit verwijzen in zijn werk gaat wordt uitgelegd in paragraaf 5.5.4.

Om aan te geven dat een objecttype bezielt is, moet de tekst “(bezielt)” achter het naamwoord en de eventuele meervoudsvorm gezet worden. In ons RegelSpraak voorbeeld ten aanzien van het objecttype *Natuurlijk persoon* ziet dat er als volgt uit:

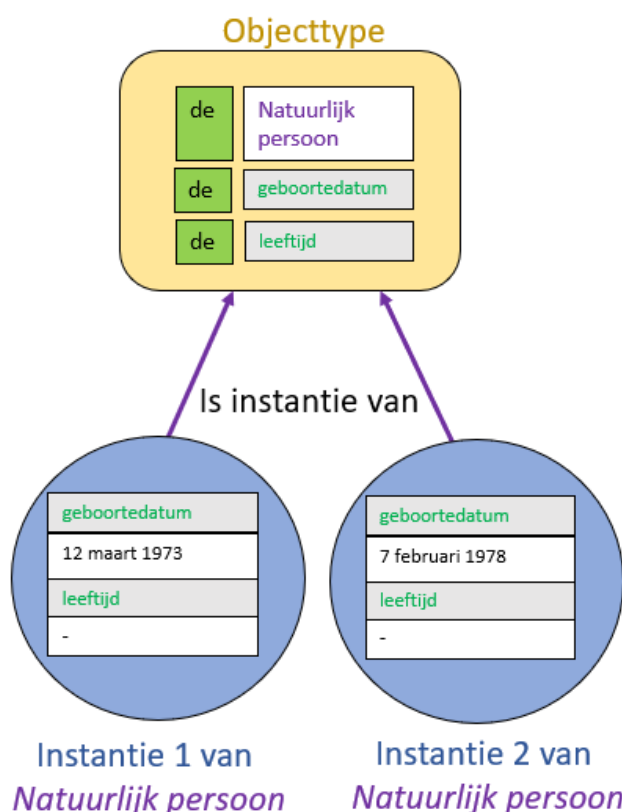
objecttype de *Natuurlijk persoon* (mv: Natuurlijke personen) (bezielt)

Syntax Objecttype specificatie:

```
"Objecttype" <naamwoord> [" (bezielt)"] \n
```

3.2 Eigenschappen van objecttypen

Objecttypen in RegelSpraak specificeren de eigenschappen die elke instantie van het objecttype kan hebben. We kunnen bij ons objecttype *Natuurlijk persoon* bijvoorbeeld de eigenschappen *geboortedatum* en *leeftijd* definiëren. Elke instantie van dit objecttype zal deze eigenschappen dan beschikbaar hebben. Zie Figuur 2 voor een illustratie hiervan.



Figuur 2: een objecttype *Natuurlijk persoon* met twee eigenschappen, de *geboortedatum* en de *leeftijd*, dat twee objecten specificeert, oftewel instanties van dit objecttype. De eigenschap *geboortedatum* heeft bij het aanmaken van de instantie een waarde gekregen, de waarde van de eigenschap *leeftijd* is nog leeg.

3.2.1 Attributen

Van de eigenschappen van objecttypen zijn verschillende soorten. Een eerste belangrijk soort eigenschap betreft het **attribuut**. Een attribuut is een gegeven waaruit een objecttype onder andere opgebouwd is en dat op enig moment een concrete waarde kan, en soms moet, hebben. Een attribuut kan dan ook gezien worden als de representatie van een dergelijk gegeven, welke bijvoorbeeld bij de implementatie in een fysiek datamodel van een relationele database vertaald wordt naar een kolom van een tabel.

Een attribuut wordt net als een objecttype aangeduid met een naamwoord (zie paragraaf 3.1) met daarna een aanduiding van het **datatype**: hiermee wordt bepaald welke soort waarden het attribuut kan aannemen. Een attribuut voor een luchthaven van vertrek van een vlucht zal bijvoorbeeld tekstwaarden bevatten (de namen van de luchthavens), maar een attribuut voor de afstand tot de bestemming van de vlucht zal getallen bevatten (het aantal kilometers tussen vertrekpunt en bestemming). Voor meer uitleg over de diverse mogelijke datatypes wordt verwezen naar paragraaf 3.3.

Een attribuut hoeft overigens geen waarde te hebben. Als het attribuut geen waarde heeft, dan spreken we van een **lege waarde**.

De attributen *geboortedatum* en *leeftijd* bij het objecttype *Natuurlijk persoon* zouden we als volgt kunnen specificeren, met de datatypes *Datum in dagen* en *Numeriek (niet-negatief geheel getal)*:

```
Objecttype de Natuurlijk persoon (mv: Natuurlijke personen) (bezield)
  de geboortedatum          Datum in dagen;
  de leeftijd                Numeriek (niet-negatief geheel getal);
```

Syntax attribuut specificatie:

Opmerking 1: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.
Opmerking 2: dimensie en tijdslijn zijn optioneel, maar mogen niet in combinatie met elkaar worden toegepast.

```
<attribuut> ::= <naamwoord> \t (<datatype> | <domeinnaam>) ["gedimensioneerd met"
<dimensienaam>] [<tijdslijn>] ";"
```

```
<datatype> ::= <numeriekdatatype> | <percentagedatatype> | <tekstdatatype> |
<booleandatatype> | <datumtijddatatype>
```

3.3 Standaarddatatypes

Bij het opstellen van RegelSpraak-regels is het bij een aantal constructies van belang om een **datatype** te specificeren. Een datatype betreft een beschrijving van de structuur waaraan een waarde, oftewel de data zelf, moet voldoen. De RegelSpraak-constructies waar een datatype van belang is, zijn attributen van objecttypen, parameters en expressies. Hiervoor geldt het volgende:

- Bij attributen van objecttypen is het belang een datatype op te geven. Hiermee wordt bepaald welke soort waarden het betreffende attribuut kan aannemen, zoals in paragraaf 3.2.1 aangegeven.
- Voor sommige regels in RegelSpraak zijn daarnaast algemene gegevens nodig die niet bij een specifiek object horen. Deze algemene gegevens worden in RegelSpraak parameters genoemd en zullen dus ook een bepaald datatype hebben. Dit wordt verder behandeld in paragraaf 3.10.
- Expressies zijn RegelSpraak constructies voor onder andere berekeningen. Zo kunnen

bijvoorbeeld numerieke waarden bij elkaar opgeteld worden, maar met tekstwaarden is dat niet mogelijk/niet zinvol. Derhalve hebben expressies ook een datatype. Dit wordt verder uitgewerkt in de hoofdstukken 5 en 6.

Hierna worden de mogelijke standaarddatatypes besproken. Elk datatype definieert hierbij een verzameling met mogelijke waarden. Die mogelijke waarden zijn allemaal van dezelfde soort. Bij een *Numeriek* datatype bestaat die verzameling bijvoorbeeld uit alle mogelijke **rationele getallen**, terwijl bij een *Boolean* datatype (een datatype gebruikt voor logica) de verzameling uit slechts twee logische elementen bestaat: *waar* en *onwaar*.

Het is verder mogelijk om de default-verzameling van een datatype verder te verkleinen. Dit verkleinen gebeurt door het datatype te **beperken**. Met een dergelijke beperking wordt een deelverzameling van de mogelijke verzameling waarden aangegeven. Als we het attribuut *leeftijd* bijvoorbeeld een datatype willen geven, dan zou het standaard *Numeriek* datatype een veel te grote verzameling opleveren. In die verzameling zitten immers ook alle negatieve getallen terwijl een leeftijd niet negatief kan zijn. Verder worden leeftijden over het algemeen uitgedrukt in hele jaren, breuken zijn in de regel dan niet nodig. Het attribuut *leeftijd* krijgt dan ook een *Numeriek* datatype dat beperkt is tot niet-negatieve gehele getallen (in de navolgende paragrafen wordt besproken hoe deze extra beperkingen toegevoegd kunnen worden):

de *leeftijd* Numeriek (niet-negatief geheel getal);

In RegelSpraak zijn vijf standaarddatatypes voorhanden, te weten Numeriek, Tekst, Boolean, Datum-tijd en Percentage. Deze zullen hierna afzonderlijk toegelicht worden met aansluitend een opsomming van diverse voorbeelden.

3.3.1 Datatype Numeriek

Een Numeriek datatype bevat rationale getallen, al dan niet uitgebreid met een eenheid.

Irrationale getallen³ en **complexe getallen**⁴ vallen buiten de verzameling numerieke waarden en kunnen niet in RegelSpraak gebruikt worden.

Bij een Numeriek datatype gelden de volgende twee nadere specificaties:

1. een Numeriek datatype heeft altijd een **verplichte specificatie** die het aantal decimalen aangeeft:
 - a. **geheel getal** (0 decimalen);
 - b. **getal met x decimalen** (hierbij moet het aantal decimalen verplicht aangegeven worden);
 - c. **getal**⁵ (onbeperkt aantal decimalen);
2. daarnaast kan aan een Numeriek datatype een **optionele specificatie** worden toegevoegd die aangeeft welk bereik het getal heeft ⁶:
 - a. **negatief**;
 - b. **positief**;
 - c. **niet-negatief**;
 dit is gelijk aan *positief* maar dan inclusief het getal 0.

In RegelSpraak is dit als volgt weer te geven (tekst tussen haakjes na de datatypevermelding is toegevoegd):

³ Zie https://nl.wikipedia.org/wiki/Irrationaal_getal

⁴ Zie https://nl.wikipedia.org/wiki/Complex_getal

⁵ Hierbij komen in basis alleen maar rationale getallen voor maar de benaming ("getal") is algemeen gehouden.

⁶ Op basis van deze drie optionele beperkingen zou de optie "niet-positief" (gelijk aan *negatief* maar dan inclusief het getal 0) ook verwacht worden, maar deze is in de praktijk nog niet nodig geweest. Vandaar de afwezigheid van deze optie.

Objecttype de **Natuurlijk persoon** (mv: Natuurlijke personen) (beziëld)
 het **identificatienummer** Numeriek (positief geheel getal);
 de **leeftijd** Numeriek (niet-negatief geheel getal);
 de **te betalen belasting** Numeriek (getal met 2 decimalen)

Objecttype de **vlucht** (mv: vluchten)
 de **afstand tot bestemming** Numeriek (geheel getal);

Syntax Numeriek datatype specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```
<numeriekdatatype> ::= "Numeriek (" <getalspecificatie> ")" ["met eenheid"
[(<eenheidmacht>+ | "1") "/" ](<eenheidmacht>+)]

<getalspecificatie> ::= ["negatief" | "niet-negatief" | "positief"] ("geheel getal" |
"getal met" <aantaldecimalen> "decimalen" | "getal")

<aantaldecimalen> ::= <positiefgeheelgetal>

<eenheidmacht> ::= <eenheidsafkorting>[^(<exponent>)]

<exponent> ::= <geheelgetal>
```

3.3.2 Datatype Tekst

Een **Tekst** datatype bevat de meeste soorten letters, tekens en woorden die in een tekst kunnen voorkomen. Alle letters en tekens die in de Unicode standaard⁷ zijn opgenomen, kunnen hierbij worden gebruikt. Een waarde met datatype *Tekst* staat hierbij altijd tussen aanhalingstekens. Een voorbeeld is het attribuut *luchthaven van vertrek* met als waarde “Amsterdam Schiphol”. Een waarde met een *Tekst* datatype kan hierbij willekeurig lang zijn⁸ en het datatype kan niet verder worden beperkt. In RegelSpraak is het definiëren van een tekstattribuut als volgt mogelijk:

Objecttype de **Natuurlijk persoon** (mv: Natuurlijke personen) (beziëld)
 het **burgerservicenummer** Tekst;

Syntax Tekst datatype specificatie:

```
<tekstdatatype> ::= "Tekst"
```

3.3.3 Datatype Boolean

Een **Boolean** datatype kan precies twee waarden aannemen:

1. **waar**;
2. **onwaar**.

Een voorbeeld is het attribuut *bereikbaar per trein*. Een Boolean datatype kan daarnaast niet verder worden beperkt. In RegelSpraak is het definiëren van een Boolean attribuut als volgt mogelijk:

Objecttype de **vlucht** (mv: vluchten)
bereikbaar per trein Boolean;
gebruik fossiele brandstof minder dan 50 procent Boolean;

Syntax Boolean datatype specificatie:

```
<booleandatatype> ::= "Boolean"
```

⁷ Zie <https://nl.wikipedia.org/wiki/Unicode>

⁸ In de praktijk zal de lengte onderhevig zijn aan technische beperkingen

3.3.4 Datatype Datum-tijd

Een **Datum-tijd** datatype wordt gebruikt voor attributen met datums en eventueel tijden/tijdstippen. Een voorbeeld is *geboortedatum* of *vluchtdatum*. Een Datum-tijd attribuut heeft één nadere beperking die aangeeft hoe nauwkeurig een waarde moet zijn:

- in dagen;
- in millisecondes.

Hierbij betekent *in dagen* een normale datumwaarde (bijvoorbeeld 12-03-2023) en *in millisecondes* een datumwaarde aangevuld met tijdwaarde (bijvoorbeeld 12-03-2023 11:34:42.679).

In RegelSpraak is dit als volgt weer te geven:

Objecttype de *Natuurlijk persoon* (mv: Natuurlijke personen) (bezielt)
de *geboortedatum* Datum in dagen;

Objecttype de *Vlucht* (mv: Vluchten)
de *vluchtdatum* Datum in dagen;
de *verwachte datum-tijd van aankomst* Datum en tijd in millisecondes;
de *verwachte datum-tijd van vertrek* Datum en tijd in millisecondes;

Syntax Datum (en tijd) specificatie:

```
<datumtijddatatype> ::= "Datum in dagen" | "Datum en tijd in millisecondes"
```

3.3.5 Datatype Percentage

Een **Percentage** datatype wordt gebruikt voor attributen met een percentagewaarde. Een doorgaans bekend voorbeeld hierbij is het btw percentage.

Een Percentage datatype is in basis een gespecialiseerd numeriek datatype. Dat betekent dat ook bij een Percentage attribuut altijd een verplichte beperking moet worden opgegeven die aangeeft wat voor soort getal het is, en mogelijk een optionele beperking die aangeeft welk bereik het getal heeft. Voor meer informatie hierover wordt verwezen naar het Numeriek datatype eerder in dit hoofdstuk.

Een *Percentage* attribuut heeft verder altijd als eenheid %. Het is echter mogelijk om dit nader te specificeren, bijvoorbeeld als %/jaar (procent per jaar). Voor meer informatie over eenheden wordt verwezen naar paragraaf 3.7.

In RegelSpraak kan een Percentage datatype als volgt gebruikt worden ⁹:

Parameter het *percentage reisduur tweede schijf* : Percentage (geheel getal)

⁹ In dit voorbeeld is een parameter opgenomen aangezien in de TOKA-casus verder geen attributen voorkomen met het datatype Percentage. Voor meer informatie over parameters wordt verwezen naar paragraaf 0.

Syntax Percentage specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```
<percentagedatatype> ::= "Percentage (" <getalspecificatie> ")" ["met eenheid %" ["/" <eenheidsafkorting>]]
```

3.3.6 Voorbeelden bij standaarddatatypes

In onderstaande tabel staan de standaarddatatypes samengevat, inclusief toegestane en niet-toegestane voorbeelden.

Datatype	Nadere specificaties	Toegestane waarden voorbeelden	Niet toegestane waarden voorbeelden ¹⁰
Numeriek	<ul style="list-style-type: none"> geheel getal negatief geheel getal niet-negatief geheel getal positief geheel getal getal met x decimalen (bijv. x=2) negatief getal met x decimalen (bijv. x=2) niet-negatief getal met x decimalen (bijv. x=2) positief getal met x decimalen (bijv. x=2) getal negatief getal niet-negatief getal positief getal 	<ul style="list-style-type: none"> -49; 0; 12; -49; 0; 12; 12; -12,34; 0; 34,56; -49; 12; -12,34; 0; 34,56; 34,56; -3/8; 0; 2/7; $\frac{3}{8}$; 0; $\frac{2}{7}$; $\frac{2}{7}$; 	<ul style="list-style-type: none"> -12,34; 34,56; TRUE; onwaar; "Jan Jansen"; 30-10-2022; 12%; 0; 12; -49; -49; 0; 12,345; TRUE; onwaar; "Jan Jansen"; 30-10-2022; 12%; 0; 34,56; -12,345; -12,34; 34,567; 0; -12,34; 34,567; TRUE; onwaar; "Jan Jansen"; 30-10-2022; 12%; 0; $\frac{2}{7}$; $\frac{1}{8}$; 0; $\frac{1}{8}$;
Tekst		<ul style="list-style-type: none"> "Jan Jansen" "Amsterdam" 	<ul style="list-style-type: none"> TRUE; FALSE; waar; onwaar; <i>Alles wat niet tussen quotes staat;</i>
Boolean		<ul style="list-style-type: none"> waar onwaar 	<ul style="list-style-type: none"> TRUE; FALSE; "Jan Jansen";

¹⁰ Bij de niet-toegestane voorbeelden worden de voorbeelden die in basis een verkeerd datatype hebben slechts eenmalig benoemd, en dus niet bij de onderverdeling n.a.v. "negatief", "niet-negatief" en "positief".

			<ul style="list-style-type: none"> • 12; • 34,56; • 29-10-2022; • -12%; • 5%;
Datum-tijd	<ul style="list-style-type: none"> • in dagen • in millisecondes 	<ul style="list-style-type: none"> • 30-10-2022; • 29-10-2022 02:04:06.123; 	<ul style="list-style-type: none"> • TRUE; • FALSE; • "Jan Jansen"; • -7 • 12; • 34,56; • waar; • onwaar; • -12%; • 5%; • 30-02-2023 • 25:12:56.667
Percentage	<ul style="list-style-type: none"> • geheel getal • negatief geheel getal • niet-negatief geheel getal • positief geheel getal • getal met x decimalen (x=3) • negatief getal met x decimalen (x=3) • niet-negatief getal met x decimalen (x=3) • positief getal met x decimalen (x=3) • getal • negatief getal • niet-negatief getal • positief getal 	<ul style="list-style-type: none"> • -12%; 0%; 5%; • -12%; • 0%; 5%; • 5%; • -2,679%; 0,000%; 3,142%; -12,34%; 34%; • -2,679%; • 0%; 3,142%; • 3,142%; • $-\frac{1}{8}\%$; 0%; $\frac{1}{7}\%$; • $-\frac{1}{8}\%$; • 0%; $\frac{1}{7}\%$ • $\frac{1}{7}\%$ 	<ul style="list-style-type: none"> • -12,34%; 34,56%; TRUE; onwaar; "Jan Jansen"; 30-10-2022; • 0%; 5%; • -12%; • -12%; 0%; • 12,3456%; TRUE; onwaar; "Jan Jansen"; 30-10-2022; • 0%; 3,142%; 12,3456%; • -2,679%; 12,3456%; • -2,679%; 0%; 12,3456%; • TRUE; onwaar; "Jan Jansen"; 30-10-2022; • 0%; $\frac{1}{7}\%$; • $-\frac{1}{8}\%$; • 0%; $-\frac{1}{8}\%$

Tabel 2: de vijf standaarddatatypes in RegelSpraak.

3.4 Domeinen

In RegelSpraak is het mogelijk om naast de standaarddatatypes zoals beschreven in de voorgaande paragraaf een **domein** te specificeren als zijnde het datatype voor een attribuut (overigens ook voor een parameter, zie verder paragraaf 3.10). Een domein wordt geïdentificeerd door een unieke naam en is qua inhoud een gespecificeerde verzameling van mogelijke waarden. Deze gespecificeerde verzameling kan gebaseerd zijn op een

standaarddatatype (met uitzondering van de boolean), al dan niet met beperking, of het kan een lijst met mogelijke enumeratiewaarden zijn. Deze twee opties worden hierna in subparagrafen aan de hand van een voorbeeld verder uitgelegd.

3.4.1 Domein op basis van een standaarddatatype

In de TOKA-casus spelen geldbedragen een rol op verschillende plekken. Een geldbedrag is vaak, in ieder geval in het kader van de TOKA-casus, een getal met 2 decimalen. Het is daarom mogelijk om een domein *Bedrag* te specificeren met als datatype *Numeriek* en met nadere beperking *getal met 2 decimalen*. We kunnen daarna alle attributen die een bedrag representeren als datatype het domein *Bedrag* geven en geven daarmee aan dat elk bedrag numeriek is met 2 decimalen.

Op deze manier kunnen definities van objecttypen begrijpelijker worden gemaakt en kan een vaak voorkomende soort waarde (zoals een bedrag) op een centrale plek eenduidig gedefinieerd worden. In RegelSpraak kan het genoemde domein *Bedrag* als volgt gespecificeerd worden:

Domein *Bedrag* is van het type *Numeriek* (getal met 2 decimalen) met eenheid €

In de specificatie van het objecttype *Natuurlijk persoon* kan dit domein vervolgens gebruikt worden, bijvoorbeeld voor het specificeren van het attribuut *te betalen belasting*. In RegelSpraak ziet dat er als volgt uit (een domein wordt cursief weergegeven):

Objecttype *de Natuurlijk persoon* (mv: Natuurlijke personen) (bezield)
 de *leeftijd* Numeriek (niet-negatief geheel getal);
 de *te betalen belasting* *Bedrag*;

3.4.2 Domein op basis van een enumeratie

In plaats van het gebruik van een van de standaarddatatypes bij de definitie van een domein kan een domein ook gespecificeerd worden met een zogenaamde **enumeratie**. Een enumeratie is een vooraf-gedefinieerde uitputtende lijst met waarden die een attribuut of parameter kan aannemen. Een voorbeeld is het attribuut *luchthaven van vertrek*: er is binnen de TOKA-casus slechts een beperkt aantal luchthavens waarvandaan een vlucht kan vertrekken. Deze luchthavens kunnen expliciet benoemd worden in een (enumeratie)lijst. Deze lijst kunnen we als domein specificeren en attributen met dit domein kunnen dan alleen een waarde uit deze lijst aannemen. Een dergelijke enumeratie ziet er in RegelSpraak als volgt uit:

Domein *Luchthavens* is van het type *Enumeratie*

'Amsterdam Schiphol'
 'Groningen Eelde'
 'Parijs Charles de Gaulle'
 'Londen Heathrow'

Syntax Domein definitie:

```
<domeindefinitie> ::= "Domein" <domeinnaam> "is van het type" (<datatype> |
<enumeratiespecificatie>)

<enumeratiespecificatie> ::= "Enumeratie" \n (\t <enumeratiewaarde> \n)+

<domeinnaam> ::= <karakterreeks>
```

3.5 Kenmerken

Naast attributen kunnen objecttypen ook een ander soort eigenschap bevatten, namelijk zogenaamde **kenmerken**. Een kenmerk is een eigenschap die een object wel of niet heeft. Default zal een kenmerk er niet zijn (onwaar), en alleen als het kenmerk in de invoer wordt gezet of als er een kenmerktoekenningregel (zie paragraaf 9.2) toepasbaar is, kan het betreffende kenmerk waar zijn. Een kenmerk lijkt hierdoor op een Boolean attribuut (zie paragraaf 3.2.1) maar het is net iets anders:

1. Je zou bijvoorbeeld een attribuut met datatype Boolean *minderjarig* kunnen hebben. Dit attribuut krijgt de waarde *waar* als de *leeftijd* van de instantie kleiner is dan 18 jaar en de waarde *onwaar* als dat niet het geval is. Zie Figuur 3 voor een illustratie hiervan waarbij dit verwoord kan worden als “indien minderjarig van de persoon gelijk is aan waar”.



Figuur 3: drie instanties van objecttype *Natuurlijk persoon* met allemaal het Boolean attribuut *minderjarig*.

2. Je zou echter ook een kenmerk *minderjarig* kunnen hebben. Dit kenmerk zal aanwezig zijn bij instanties waarvan de *leeftijd* kleiner is dan 18 jaar. Zie Figuur 4 voor een illustratie hiervan waarbij dit verwoord kan worden als “indien de persoon minderjarig is”.



Figuur 4: drie instanties van objecttype *Natuurlijk persoon* waarvan één het kenmerk *minderjarig* heeft.

Het verschil tussen een attribuut *minderjarig* en een kenmerk *minderjarig*, zoals hierboven toegelicht en geïllustreerd, is dus vooral terug te zien in hoe beide zaken in RegelSpraak-regels worden weergegeven. Bij Boolean attributen spreken we over een instantie waarvan attribuut *minderjarig* altijd aanwezig is met een de waarde die waar of onwaar is of (nog) onbekend (leeg), terwijl we bij kenmerken spreken over een instantie die *minderjarig* is indien dat het geval is. Dit betekent concreet dat een kenmerk in tegenstelling tot een attribuut nooit leeg kan zijn: als een kenmerk van toepassing is, dan is dat zo en is het kenmerk aanwezig. Er wordt dan ook niet getest op waarde maar op aanwezig zijn. Om de leesbaarheid van RegelSpraak-regels te bevorderen wordt geadviseerd om kenmerken te gebruiken waar dat mogelijk is in plaats van Boolean attributen.

Een kenmerk wordt gespecificeerd bij het desbetreffende objecttype. De specificatie bestaat uit een naamwoord, gevolgd door het woord *kenmerk*. Een kenmerk kan daarna verder gespecificeerd worden als *bezittelijk* of *bijvoeglijk*, waarbij de volgende toelichting van toepassing is.

1. Een *bijvoeglijk* kenmerk is een kenmerk waarover we spreken met het werkwoord “zijn”. Een persoon kan bijvoorbeeld minderjarig zijn. Voor ons objecttype *Natuurlijk persoon* specificeren we dat als een *bijvoeglijk* kenmerk “minderjarig”. Bij bijvoeglijke kenmerken zijn geen lidwoorden nodig: we zeggen in dat geval *Natuurlijk persoon is minderjarig*. Er hoeft dan ook geen lidwoord gespecificeerd te worden in het naamwoord voor het kenmerk, wel moet het woord *is* voor het naamwoord worden gezet. In RegelSpraak ziet dit er als volgt uit:

is *minderjarig* kenmerk (bijvoeglijk);

2. Een *bezittelijk* kenmerk is een kenmerk waarover we spreken met het werkwoord “hebben”. Een persoon kan bijvoorbeeld een bepaald recht hebben. In het kader van de TOKA-casus hebben personen al dan niet het recht op duurzaamheidskorting. Dit kunnen we specificeren als een bezittelijk kenmerk *het recht op duurzaamheidskorting* van objecttype *Natuurlijk persoon*. In RegelSpraak-regels zal dan *Natuurlijk persoon heeft het recht op duurzaamheidskorting* gebruikt worden.

het *recht op duurzaamheidskorting* kenmerk (bezittelijk);

3. Een kenmerk kan daarnaast ook noch bijvoeglijk, noch bezittelijk zijn. In dat geval spreken we erover met het werkwoord “zijn” en een lidwoord. In het kader van de TOKA-casus moet er bijvoorbeeld onderscheid gemaakt worden tussen niet-minderjarige passagiers met een leeftijd boven of onder de 24 jaar. Dit kunnen we specificeren als een kenmerk *de passagier van 18 tot en met 24 jaar*. In RegelSpraak-regels zal dan *Natuurlijk persoon is een passagier van 18 tot en met 24 jaar* gebruikt worden.

de *passagier van 18 tot en met 24 jaar* kenmerk;

Als deze drie soorten kenmerken gespecificeerd worden bij het objecttype *Natuurlijk persoon*, dan kan dat er als volgt uit zien:

```
Objecttype de Natuurlijk persoon (mv: Natuurlijke personen)(beziend)
  is minderjarig kenmerk (bijvoeglijk);
  het recht op duurzaamheidskorting kenmerk (bezittelijk);
  de passagier van 18 tot en met 24 jaar kenmerk;
```

Na het specificeren van een kenmerk moet er ook een regel worden opgesteld die bepaalt of een instantie van het objecttype het kenmerk moet krijgen, een zogenaamde **kenmerktoekenning**. Meer informatie hierover is terug te vinden in paragraaf 9.2.

```

Syntax kenmerk specificatie:<kenmerk> ::= ((<naamwoord> "kenmerk") | <bezittelijkkenmerk>
| <bijvoeglijkkenmerk>) [<tijdslijn>] ";"

<bezittelijkkenmerk> ::= <naamwoord> "kenmerk (bezittelijk) "

<bijvoeglijkkenmerk> ::= "is" <naam> ["(mv: " <meervoudsvorm> ")"] "kenmerk
(bijvoeglijk) "

```

3.6 Dimensies


Elk attribuut van een objecttype kan op hetzelfde moment meerdere waarden hebben door gebruik te maken van dimensies. Een dimensie is een manier om gestructureerd meerdere beweringen over hetzelfde attribuut vast te leggen. Het inkomen van een persoon zal bijvoorbeeld niet elk jaar hetzelfde zijn, en het zou hierbij niet handig zijn om voor elk jaarinkomen een afzonderlijk attribuut te moeten specificeren. In plaats daarvan kan echter het attribuut *inkomen* met een dimensie *jaardimensie* gespecificeerd worden. Daarna kan dan het enkele attribuut *inkomen* gebruikt worden voor de verschillende inkomens in de verschillende jaren: bij gebruik van *inkomen* in een regel moeten we aangeven welk inkomen uit welk jaar er wordt bedoeld.

Een attribuut kan meerdere dimensies hebben. Een inkomen kan bijvoorbeeld naast voornoemde jaardimensie ook een netto- of een bruto-inkomen zijn. Om dat vast te leggen kunnen we een tweede dimensie toevoegen aan het attribuut inkomen. Bij gebruik van dat attribuut dienen dan altijd alle dimensies te worden aangegeven aangezien het anders niet duidelijk is welke specifieke waarde bedoeld wordt. Als we het bijvoorbeeld hebben over het inkomen van de persoon van vorig jaar, dan is het niet duidelijk of dat het netto- of bruto-inkomen is. We moeten dus expliciet aangegeven of het, het netto- dan wel bruto-inkomen van vorig jaar betreft.

Dimensies kunnen op twee manieren in RegelSpraak-regels worden weergegeven.

1. De eerste manier is *na het attribuut met voorzetsel* waarbij gekozen kan worden tussen voorzetsels *van*, *in*, *voor*, *over*, *op*, *bij* en *uit*. Bij verwijzing naar het attribuut wordt dan eerst het attribuut genoemd, dan het gedefinieerde voorzetsel, en dan het label van de dimensie. Als we bijvoorbeeld het voorzetsel *van* definiëren voor de dimensie over jaren, dan spreken we over het *inkomen van huidig jaar*.
2. De tweede manier is *voor het attribuut als bijvoeglijk naamwoord*. Het label van de dimensie wordt dan direct voor het attribuut geplaatst. Als we dit bijvoorbeeld definiëren voor de dimensie over bruto en netto, dan spreken we over het *bruto inkomen*.

Natuurlijk persoon



inkomen	bruto	netto
vorig jaar	30 000	20 000
huidig jaar	33 000	21 000

Figuur 5: een *Natuurlijk persoon* instantie, met het attribuut *inkomen* met twee dimensies.

Een dimensie bestaat uit een verzameling **labels**. Met een label geef je aan welke waarde van het attribuut moet worden gebruikt. Als we bijvoorbeeld een dimensie willen specificeren voor inkomens in verschillende jaren, dan kunnen we de labels *huidig jaar* en *vorig jaar* gebruiken. In RegelSpraak ziet dat er als volgt uit:

Dimensie de **jaardimensie**, bestaande uit de **jaardimensies** (na het attribuut met voorzetsel van):

1. vorig jaar
2. huidig jaar

De volgorde van de labels is hierbij van belang. Bij gebruik van dimensies in RegelSpraak kan namelijk verwezen worden naar een interval van labels. Bij het specificeren van een dimensie wordt de volgorde van labels bepaald door een nummer voor het label te zetten (hierboven is te zien dat label *vorig jaar* gekoppeld is aan nummer 1).

We willen echter ook een dimensie toevoegen om onderscheid te maken tussen het bruto- en netto-inkomen, dit keer een dimensie die voor het attribuut zonder voorzetsel gebruikt moet worden:

Dimensie de **brutonettodimensie**, bestaande uit de **brutonettodimensies** (voor het attribuut zonder voorzetsel):

- bruto
- netto

Bij het specificeren van het attribuut *inkomen* geven we vervolgens aan, na het specificeren van het datatype, dat deze dimensies gebruikt moeten worden bij dit attribuut:

het **inkomen** Numeriek (geheel getal) gedimensioneerd met **jaardimensie** en **brutonetodimensie**;

Het attribuut *inkomen* heeft nu eigenlijk vier mogelijke waarden (zie Figuur 5). Als we naar dit attribuut willen verwijzen, dan moeten we altijd beide dimensies gebruiken omdat anders niet duidelijk is welke van de vier waarden bedoeld wordt. Zo zal het *bruto inkomen* van een *Natuurlijk persoon* twee verschillende waarden hebben. In RegelSpraaak-regels gebruiken we daarom in dit geval het *bruto inkomen van huidig jaar* of het *bruto inkomen van vorig jaar*.

Syntax Dimensie specificatie:

```

<dimensie> ::= "Dimensie" <bepaaldlidwoord> <dimensienaam> ", bestaande uit de "
<dimensienaammeervoud> <voorzetselspecificatie> \n (<labelwaardespecificatie> \n)+

<voorzetselspecificatie> ::= ("na het attribuut met voorzetsel" ( "van" | "in" | "voor"
| "over" | "op" | "bij" | "uit" ) "):" | "(voor het attribuut zonder voorzetsel):"

<dimensienaam> ::= <karakterreeks>
<dimensienaammeervoud> ::= <karakterreeks>
<labelwaardespecificatie> ::= <digit>+ "." " <dimensiewaarde>
<dimensiewaarde> ::= <karakterreeks>

```

3.7 Eenheden en eenheidssystemen

Overall waar het numeriek datatype een rol speelt, zijn eenheden mogelijk van toepassing. Zo kunnen ook een numeriek attribuut en een numerieke parameter (zie paragraaf 3.10) een eenheid hebben als onderdeel van het betreffende gespecificeerde datatype. Een afstand kan bijvoorbeeld in meters of in kilometers zijn en een leeftijd wordt meestal uitgedrukt in jaren. Bij berekeningen moet vervolgens wel rekening gehouden worden met eenheden. Zo is het niet logisch en dus niet mogelijk om meters bij jaren op te tellen.

Voordat een eenheid toegekend kan worden aan een attribuut of parameter moet deze eerst gespecificeerd worden. Een eenheid wordt gespecificeerd met een **eenheidssysteem**: een eenheidssysteem beschrijft de te gebruiken basiseenheden door middel van een naamwoord, eventueel hun afkorting en/of symbool, en eventueel een **omrekeningspecificatie** waarmee de basiseenheid omgerekend kan worden in een andere basiseenheid uit hetzelfde eenheidssysteem.

Als we bijvoorbeeld de eenheid Euro willen gaan gebruiken, dan moeten we een eenheidssysteem specificeren waarin de euro een basiseenheid is. Verder kunnen we dan specificeren dat de euro afgekort wordt tot EUR en het symbool € heeft. Dat doen we als volgt in RegelSpraak ^{11 12}:

Eenheidssysteem valuta
de euro (mv: euros) EUR €

Als we de basiseenheid hebben gespecificeerd in een eenheidssysteem, dan kunnen we hem toekennen aan een attribuut of parameter. Dit doen we door *met eenheid* achter het datatype of de domeinnaam te zetten, gevolgd door de naam van de basiseenheid. In RegelSpraak ziet dat er als volgt uit bij een attribuut:

de **te betalen belasting** *Bedrag met eenheid euro;*

Samengestelde eenheden kunnen ook gespecificeerd worden. De basiseenheden uit de eenheidssystemen kunnen immers door middel van delingen of vermenigvuldigingen gecombineerd worden. Een snelheid heeft bijvoorbeeld een samengestelde eenheid, veelal het aantal kilometers gedeeld door het aantal uren waardoor kilometers per uur als samengestelde eenheid ontstaat. In RegelSpraak kunnen we dit specificeren door een deelstreepje "/" te zetten tussen beide basiseenheden. In RegelSpraak ziet dat er als volgt uit bij een attribuut:

de **snelheid** *Numeriek (getal) met eenheid km/u*

¹¹ Alhoewel eenheidssysteem volgens de Nederlandse spelling met twee s-en geschreven wordt, wordt in RegelSpraak slechts 1 s gebruikt.

¹² RegelSpraak bevat standaard al een eenheidssysteem voor verschillende valuta, *Valuta ISO-4217*. Het getoonde voorbeeld is slechts bedoeld om het specificeren van een eenheidssysteem te illustreren.

Verder kunnen ook machten gebruikt worden bij het samenstellen van eenheden. Een versnelling heeft bijvoorbeeld de eenheid “meter per seconde kwadraat”. In RegelSpraak kan dat als volgt gespecificeerd worden bij een attribuut:

Numeriek (getal) met eenheid m/s²

In de vorige twee voorbeelden werden twee verschillende basiseenheden voor afstand gebruikt, namelijk de meter (*m*) en de kilometer (*km*). Deze twee basiseenheden kunnen in elkaar omgerekend worden: een kilometer is gelijk aan 1000 meter. Hetzelfde geldt voor de gebruikte basiseenheden voor tijd: een uur is 3600 secondes. Dit kan bij een eenheidssysteem in RegelSpraak ook gespecificeerd worden door het opgeven van een omrekeningspecificatie. Een omrekeningspecificatie specificeert voor een basiseenheid een **omrekenfactor** waarmee de betreffende basiseenheid omgerekend kan worden in een andere basiseenheid uit hetzelfde eenheidssysteem. De omrekenfactor bestaat uit een getal, eventueel een deelstreepje, en de basiseenheid waarop de omrekenfactor betrekking heeft. Een eenheidssysteem voor afstand zouden we bijvoorbeeld als volgt kunnen specificeren:

Eenheidssysteem afstand

de millimeter (mv: millimeters)	mm	=	$\frac{\quad}{1000}$ m
de centimeter (mv: centimeters)	cm	=	$\frac{\quad}{100}$ m
de meter (mv: meters)	m		
de kilometer (mv: kilometers)	km	=	1000 m

Bij het specificeren van omrekenfactoren is het van belang dat de basiseenheden op volgorde van toenemende grootte staan. De kleinste basiseenheid dient als eerste gespecificeerd te worden.

Omrekenpecificaties zijn optioneel. De basiseenheid *meter* heeft in bovenstaande specificatie bijvoorbeeld geen omrekenpecificatie. GegevensSprak kent echter geen speciale betekenis toe aan basiseenheden die geen omrekenfactor hebben: ze zijn evengoed omrekenbaar als een andere basiseenheid naar hen verwijst. Het is echter overbodig om bij beide basiseenheden een omrekenfactor te specificeren, mede daarom dat omrekenpecificaties optioneel zijn.

Binnen eenheidssystemen kan het voorkomen dat er basiseenheden bestaan die niet direct in elkaar kunnen worden omgerekend. Een maand heeft bijvoorbeeld geen vast aantal dagen: we kunnen daardoor maanden niet direct omrekenen in dagen. In een dergelijk geval gebruiken we dan ook geen omrekeningspecificatie.

Standaard eenheidssystemen die onderdeel zijn van GegevensSpraak, zijn:

- Het eenheidssysteem voor Valuta (gebaseerd op ISO-4217). Dit eenheidssysteem bevat geen omreken factoren.
- Het eenheidssysteem Tijd.
Hieronder de specificatie van dat eenheidssysteem.

Eenheidssysteem Tijd

de milliseconde	ms	$= \frac{1}{1000} \text{ s}$
de seconde	s	$= \frac{1}{60} \text{ minuut}$
de minuut	minuut	$= \frac{1}{60} \text{ u}$
het uur	u	$= \frac{1}{24} \text{ dg}$
de dag	dg	
de week	wk	$= 7 \text{ dg}$
de maand	mnd	
het kwartaal	kw	$= 3 \text{ mnd}$
het jaar	jr	$= 12 \text{ mnd}$

Syntax Eenheidssysteem specificatie:

```

<eenheidssysteem> ::= "Eenheidssysteem" <eenheidssysteemnaam>
(\n <naamwoord> <eenheidsafkorting> [<omrekenspecificatie>]) +

<omrekenspecificatie> ::= "=" ["1/" ] <geheelgetal> <eenheidsafkorting>

<eenheidssysteemnaam> ::= <karakterreeks>

<eenheidsafkorting> ::= <karakterreeks>

```

3.8 Tijdlijnen

Attributen, kenmerken en parameters kunnen tijdsafhankelijk zijn. Hiermee wordt bedoeld dat de waarde van een attribuut of parameter of het al dan niet hebben van een kenmerk in de tijd kan veranderen.

N.B. De waarde **kan** veranderen, maar dat **hoeft niet** het geval te zijn.

Als sprake is van tijdsafhankelijke attributen, kenmerken of parameters, dan wordt bij het betreffende element een tijdlijn opgenomen. Met die tijdlijn wordt aangegeven op welke momenten in de tijd een waarde kan veranderen.

Dit kunnen de volgende perioden op de kalender zijn zoals die in het eenheidssysteem Tijd zijn opgenomen (zie paragraaf 3.7)¹³:

Periode	Definitie
Dag	Dag op de gregoriaanse kalender
Maand	Maand op de gregoriaanse kalender (van 1 ^e dag van de maand tot 1 ^e dag van de volgende maand)
Jaar	Jaar op de gregoriaanse kalender (van 1 januari tot en met 31 december)

Als de momenten waarop een waarde kan wijzigen niet gebonden zijn aan een vaste periode op de kalender, dan wordt de tijdlijn met de periode dag gebruikt. Bij toepassing van die tijdlijn kan op iedere dag een andere waarde gelden.

Hieronder voorbeelden van een specificatie van het attribuut *te betalen belasting* (met het domein Bedrag) als dat per jaar een andere waarde kan hebben en van het kenmerk recht op

¹³ Voor een volgende versie is voorzien dat ook perioden van een jaar, kwartaal en maand kunnen worden gebruikt die niet precies samenvallen met de kalender, maar een afwijkende periode betreffen. Bijvoorbeeld een jaar dat start op 15 april en loopt tot 15 april van het volgende jaar. Deze perioden krijgen dan een specifieke naam, bijvoorbeeld boekjaar.

duurzaamheidskorting dat op variabele perioden kan gelden:

de te betalen belasting	<i>Bedrag</i>	voor elk jaar;
het recht op duurzaamheidskorting	kenmerk (bezittelijk)	voor elke dag;

Syntax Tijdlijn specificatie:

```
<tijdlijn> ::= "voor" ("elke dag" | "elke maand" | "elk jaar")
```

3.9 Voorbeeld TOKA-casus: de objecttypen *Natuurlijk persoon* en *Vlucht* volledig gespecificeerd

De TOKA-casus gaat over mensen (ook wel natuurlijke personen genoemd) die een vliegreis maken. In de RegelSprak uitwerking van deze casus zijn daarom de objecttypen *Natuurlijk persoon* en *Vlucht* gedefinieerd. Binnen de TOKA-casus heeft het objecttype *Natuurlijk persoon* de volgende attributen:

1. *het identificatienummer;*
2. *de geboortedatum;*
3. *de leeftijd;*
4. *de belasting op basis van afstand;*
5. *de belasting op basis van reisduur;*
6. *de te betalen belasting;*
7. *de treinmiles op basis van evenredige verdeling;*
8. *de maximaal te ontvangen treinmiles bij evenredige verdeling volgens rangorde;*

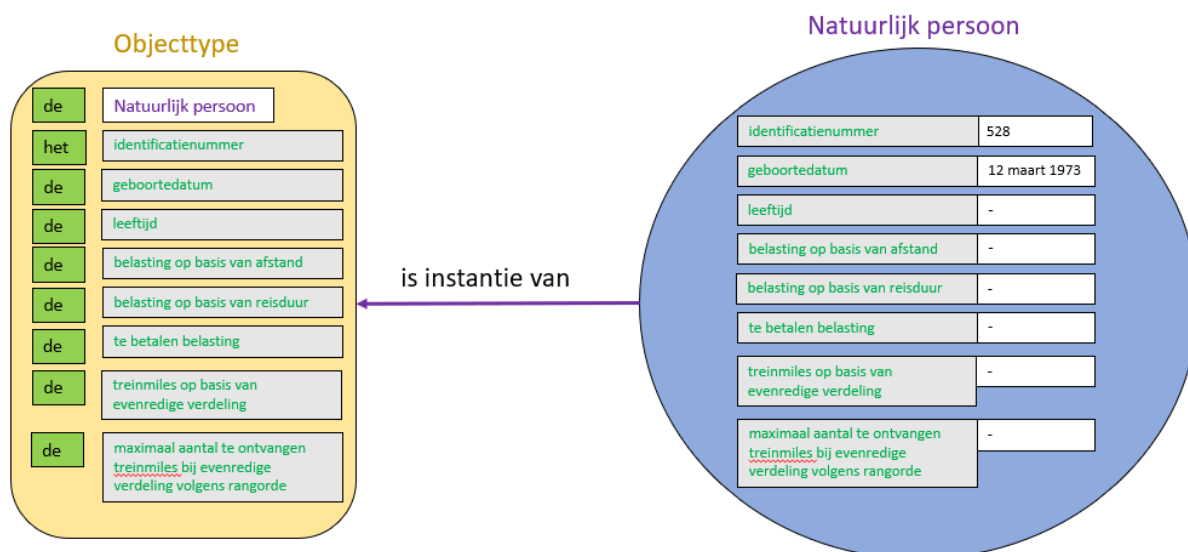
en de volgende kenmerken:

9. *is minderjarig;*
10. *is passagier van 18 tot en met 24 jaar;*
11. *is passagier van 25 tot en met 64 jaar;*
12. *is passagier van 65 jaar of ouder;*
13. *het recht op duurzaamheidskorting.*

Voor elke instantie van dit objecttype kunnen deze attributen en dientengevolge de genoemde kenmerken een andere waarde krijgen. Er kan bijvoorbeeld

1. een *Natuurlijk persoon*-object bestaan met voor het attribuut *identificatienummer* de waarde 528 en voor het attribuut *geboortedatum* de waarde 12 maart 1973;
2. een *Natuurlijk persoon*-object bestaan met voor het attribuut *identificatienummer* de waarde 531 en voor het attribuut *geboortedatum* de waarde 7 februari 1978.

We spreken dan in beide situaties van een instantie van het objecttype *Natuurlijk persoon*. Zie Figuur 6 voor een visualisatie van bovenstaand voorbeeld 1.



Figuur 6: objecttype *Natuurlijk persoon* en een instantie ervan.

Verder moeten de (op te stellen) RegelSpraak-regels voor de TOKA-casus waardes gaan bepalen op basis van een vliegreis. Er is daarom een tweede objecttype aangemaakt, te weten *Vlucht*. Dit objecttype omschrijft een aantal andere attributen die we in de regels nodig gaan hebben¹⁴:

- *de luchthaven van vertrek;*
- *de luchthaven van bestemming;*
- *de vluchtdatum;*
- *de afstand tot de bestemming;*
- *bereikbaar per trein;*
- *gebruik fossiele brandstof minder dan 50 procent;*
- *de reisduur per trein;*
- *de hoeveelheid passagiers;*
- *de hoeveelheid uitzonderingspassagiers;*
- *de leeftijd van de oudste passagier;*
- *de totale belasting op basis van afstand;*
- *de totale belasting op basis van reisduur;*
- *de totaal te betalen belasting;*
- *de verwachte datum-tijd van aankomst;*
- *de verwachte datum-tijd van vertrek;*
- *de verwachte duur;*
- *de datum-tijd voor het berekenen van de belasting op basis van afstand;*
- *het bevestigingstijdstip;*
- *het uiterste boekingstijdstip;*

En de volgende kenmerken worden hierbij onderkend en gebruikt:

- *is bereikbaar per trein;*
- *de gebruik fossiele brandstoffen minder dan 50%;*
- *de reis met paaskorting;*
- *de reiziger;*
- *is duurzaam;*
- *is belaste reis;*

¹⁴ Dit objecttype kan op soortgelijke manier gevisualiseerd worden als in Figuur 6. Dat is in dit document achterwege gelaten.

- *is belast;*
- *is rondvlucht;*
- *is bestemd voor minderjarigen;*

We kunnen hiermee de twee objecttypen uit paragraaf 3.1, te weten *Natuurlijk persoon* en *Vlucht*, volledig specificeren. In RegelSpraak zien deze er uit zoals hieronder weergegeven. Deze voorbeelden kunnen gezien worden als een RegelSpraak samenvatting van dit hoofdstuk zoals tot nu toe besproken.

Objecttype de *Natuurlijk persoon* (mv: Natuurlijke personen) (bezield)

is minderjarig	kenmerk (bijvoeglijk);
is passagier van 18 tot en met 24 jaar	kenmerk;
is passagier van 25 tot en met 64 jaar	kenmerk;
is passagier van 65 jaar of ouder	kenmerk;
het recht op duurzaamheidskorting	kenmerk (bezittelijk);
het identificatienummer	Numeriek (positief geheel getal);
de geboortedatum	Datum in dagen;
de leeftijd	Numeriek (niet-negatief geheel getal) met eenheid jr;
de belasting op basis van afstand	<i>Bedrag;</i>
de belasting op basis van reisduur	<i>Bedrag;</i>
de te betalen belasting	<i>Bedrag;</i>
de treinmiles op basis van evenredige verdeling	Numeriek (geheel getal);
de maximaal te ontvangen treinmiles bij evenredige verdeling volgens rangorde	Numeriek (geheel getal);

Objecttype de *Vlucht* (mv: Vluchten)

is bereikbaar per trein	kenmerk (bijvoeglijk);
de gebruik fossiele brandstoffen minder dan 50%	kenmerk (bezittelijk);
de reis met paaskorting	kenmerk;
de reiziger	kenmerk (bezittelijk);
is duurzaam	kenmerk (bijvoeglijk);
is belaste reis	kenmerk;
is belast	kenmerk (bijvoeglijk);
is rondvlucht	kenmerk;
is bestemd voor minderjarigen	kenmerk;
de luchthaven van vertrek	<i>Luchthavens;</i>
de luchthaven van bestemming	<i>Luchthavens;</i>
de vluchtdatum	Datum in dagen;
de afstand tot bestemming	Numeriek (geheel getal);
bereikbaar per trein	Boolean;
gebruik fossiele brandstof minder dan 50 procent	Boolean;
de reisduur per trein	Numeriek (geheel getal);
de hoeveelheid passagiers	Numeriek (geheel getal);
de hoeveelheid uitzonderingspassagiers	Numeriek (geheel getal);
de leeftijd van de oudste passagier	Numeriek (niet-negatief geheel getal) met eenheid jr;
de totale belasting op basis van afstand	<i>Bedrag;</i>
de totale belasting op basis van reisduur	<i>Bedrag;</i>
de totaal te betalen belasting	<i>Bedrag;</i>
de verwachte datum-tijd van aankomst	Datum en tijd in millisecondes;
de verwachte datum-tijd van vertrek	Datum en tijd in millisecondes;
de verwachte duur	Numeriek (geheel getal);
de datum-tijd voor het berekenen van de belasting op basis van afstand	Datum en tijd in millisecondes;
het bevestigingstijdstip	Datum en tijd in millisecondes;
het uiterste boekingstijdstip	Datum en tijd in millisecondes;

Domein *Bedrag* is van het type Numeriek (getal met 2 decimalen)

Domein *Luchthavens* is van het type Enumeratie

Amsterdam Schiphol
Groningen Eelde
Parijs Charles de Gaulle
London Heathrow

3.10 Parameters

Voor sommige regels in RegelSpraak zijn algemene gegevens nodig die niet bij een specifiek objecttype horen. Het percentage van het hoge btw-tarief is daar een voorbeeld van, of in het kader van de TOKA-casus de *korting bij gebruik niet-fossiele brandstof*. Dit soort gegevens gelden voor alle gevallen die op een bepaald moment in de tijd worden behandeld. In RegelSpraak kunnen we dit soort gegevens gebruiken door een **parameter** voor deze gegevens te specificeren. Een parameter is eigenlijk een soort globaal attribuut: een attribuut dat niet bij een specifiek objecttype hoort en waarvan de waarde binnen de context op elk moment bekend is. Een parameter moet echter wel altijd, net als een attribuut (zie paragraaf 3.2.1), een datatype hebben (zie paragraaf 3.3). Als het om een numeriek datatype gaat of om een percentage dan kan er ook een eenheid bij gespecificeerd worden (zie paragraaf 3.7).

Een voorbeeld van een parameter uit de TOKA-casus is zoals gezegd de *korting bij gebruik niet-fossiele brandstof*. Dit is een gegeven dat nodig is bij de RegelSpraak-regels, maar dat niet bij een *Natuurlijk persoon* of een *Vlucht* hoort. Deze informatie moet dus gespecificeerd worden als een parameter waarbij gebruik wordt gemaakt van het al gespecificeerde domein *Bedrag*:

Parameter de *korting bij gebruik niet-fossiele brandstof* : *Bedrag*

Syntax Parameter specificatie:

```
<parameterdefinitie> ::= "Parameter" <parametermetlidwoord> ":" (<datatype> |
<domeinnaam>) [<tijdlijn>] ";"

<parametermetlidwoord> ::= <bepaaldlidwoord> <parameternaam>

<parameternaam> ::= <karakterreeks>
```

N.B. in verband met beheer- en aanpasbaarheid van RegelSpraak-regels worden parameters doorgaans ook gebruikt om te voorkomen dat cijfers hard (dat wil zeggen niet aanpasbaar zonder de regel aan te passen) in de regels gezet worden. Een voorbeeld is het eerder genoemde hoge btw-tarief. Een apart construct kan hierbij gemaakt worden voor bijvoorbeeld het getal 0, denk dan aan het geforceerd gelijkstellen op 0 indien het resultaat van een berekening negatief is gebleken.

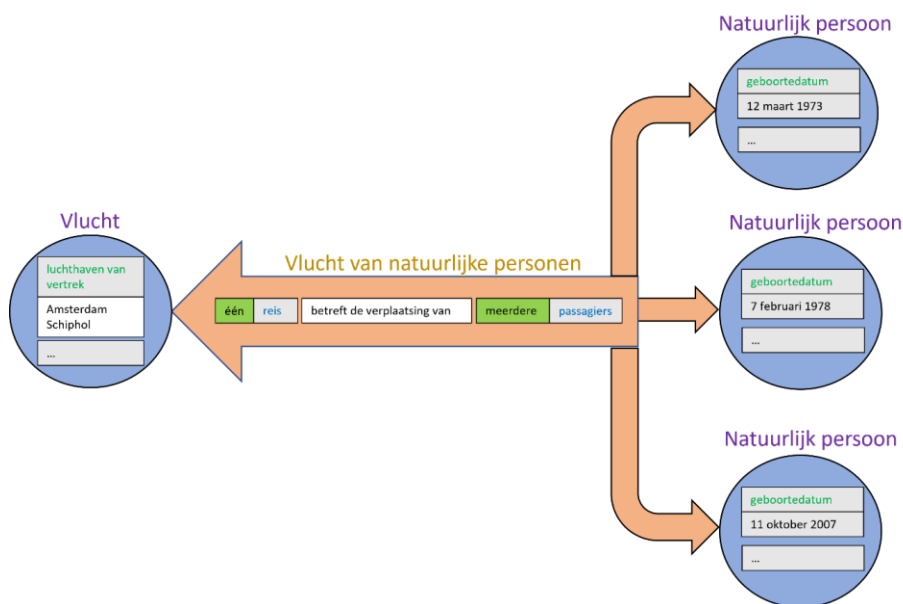
3.11 Feittypen en rollen

Er kunnen relatief zijn tussen objecttypen. In de TOKA-casus is er bijvoorbeeld een relatie tussen *Natuurlijk persoon* en *Vlucht*: een vlucht heeft nu eenmaal passagiers en die passagiers zijn natuurlijke personen. Om een relatie uit te drukken tussen twee objecttypen moet een **feittype** gespecificeerd worden. Een feittype bestaat uit **rollen** die gespeeld worden door instanties van objecttypen waaraan deze rollen verbonden zijn. Door de verschillende rollen van een feittype in te vullen met instanties van de objecttypen waarmee de rollen verbonden zijn, wordt een relatie gelegd tussen deze instanties.

Een rol kan in deze situatie **enkelvoudig** of **meervoudig** zijn. Een enkelvoudige rol kan slechts door een enkele instantie van het bijbehorende objecttype worden geïntanceerd, terwijl een meervoudige rol door een verzameling van instanties van het bijbehorende objecttype kan worden geïntanceerd. Zie verder navolgend voorbeeld ter illustratie.

In de TOKA-casus hebben we de objecttypen *Vlucht* en *Natuurlijk persoon* gedefinieerd. We moeten nu een relatie leggen tussen deze twee objecttypen, een vlucht heeft immers meerdere passagiers. Dit doen we door een feittype aan te maken met twee rollen, zie navolgend figuur.

- De eerste rol is *reis* welke geïntanceerd moet worden door een object van het objecttype *Vlucht*: dit is binnen de TOKA-casus een enkelvoudige rol.
- De tweede rol is *passagier* welke geïntanceerd moet worden door een object van het objecttype *Natuurlijk persoon*: dit is een meervoudige rol aangezien een enkele vlucht binnen de TOKA-casus meerdere passagiers kan hebben.



Figuur 7: feittype *Vlucht van natuurlijke personen* dat een instantie van objecttype *Vlucht* koppelt aan drie instanties van objecttype *Natuurlijk persoon*.

In RegelSpraak wordt een feittype gespecificeerd met het woord *Feittype*, gevolgd door de naam van het feittype. Vervolgens worden de rollen gespecificeerd:

Feittype vlucht van natuurlijke personen
 de *reis* *vlucht*
 de *passagier* *Natuurlijk persoon*
Eén reis betreft de verplaatsing van *meerdere passagiers*



Figuur 8: feittype *Vlucht van natuurlijke personen*. Het feittype omschrijft de enkelvoudige rol *Reis* die geïntanceerd wordt door objecttype *Vlucht*, en de meervoudige rol *passagiers* die geïntanceerd wordt door objecttype *Natuurlijk persoon*.

Een feittype kan verder **wederkerig** zijn: dit betekent dat de twee objecttypen dezelfde rol vervullen. Een voorbeeld hiervan is een relatie tussen twee partners. De twee personen waarvoor deze relatie geldt hebben allebei dezelfde rol, namelijk de rol van partner. Bij een wederkerig feittype kan alleen gebruik worden gemaakt van een enkelvoudige rol. In RegelSpraak wordt een wederkerig feittype als volgt gedefinieerd:

wederkerig feittype Partnerrelatie
 de *partner* *Natuurlijk persoon*
Eén partner heeft *één partner*



Figuur 9: wederkerig feittype Partnerrelatie. De enkelvoudige rol *partner* legt een relatie tussen twee instanties van objecttype *Natuurlijk persoon*.

Syntax Feittype specificatie:

```
<feittypedefinitie> ::= "Feittype" <feittypenaam> \n
[<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n
[<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n
("één" <rolnaam> | "meerdere" <meervoudrolnaam>) <relatiebeschrijving> ("één" <rolnaam>
| "meerdere" <meervoudrolnaam>)

<wederkerigfeittypedefinitie> ::= "Wederkerig feittype" <feittypenaam> \n
[<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n
(("één" <rolnaam> <relatiebeschrijving> "één" <rolnaam>) | ("meerdere" <rolnaam>
<relatiebeschrijving> "meerdere" <rolnaam>))

<feittypenaam> ::= <karakterreeks>
<rolnaam> ::= <karakterreeks>
<meervoudrolnaam> ::= <karakterreeks>
<relatiebeschrijving> ::= <karakterreeks>
```

3.12 Dagsoort

In het objectmodel kunnen **dagsoorten** gedefinieerd worden. Deze dagsoorten kunnen als eenheid van een attribuut met datatype Numeriek gebruikt worden.

Dagsoorten kunnen in regels gebruikt worden om bijvoorbeeld te controleren of een bepaalde datum een feestdag is (oftewel wanneer het een Nieuwjaarsdag, Paasdag, Koningsdag, Hemelvaartsdag, Pinksterdag of Kerstdag is), of om bijvoorbeeld een aantal werkdagen op te tellen bij een datum (zo kunnen feestdagen worden overgeslagen bij het optellen van dagen bij een datum).

Dagsoorten kunnen in het objectmodel gespecificeerd worden door een lidwoord, de naam van de dagsoort en een meervoud te specificeren. Een specificatie van een Kerstdag kan er dan als volgt uitzien:

Dagsoort de kerstdag (mv: kerstdagen)

Syntax Dagsoort specificatie:

```
<dagsoort> ::= "Dagsoort" <naamwoord>
```

In een **dagsoortdefinitie** worden voorwaarden opgenomen om concreet te bepalen wanneer een datum van dat dagsoort is. Zie hiervoor verder paragraaf 9.8.

4 De RegelSpraak-regels

RegelSpraak-regels zijn opgebouwd volgens een vast patroon. Dit hierna in detail besproken patroon bestaat uit

1. een **regel** (zie paragraaf 4.1),
2. een (of meerdere) **regelversie(s)** (zie paragraaf 4.2) en
3. een **basispatroon** (zie paragraaf 4.3).

Een compleet voorbeeld behorende bij het vaste patroon van een RegelSpraak-regel wordt uitgewerkt in paragraaf 4.4.

4.1 Regel

Allereerst zal een RegelSpraak-regel een naam moeten krijgen. De naamgevingsconstructie van een RegelSpraak-regel begint altijd met het (vaste) woord *Regel*, gevolgd door een zelf te kiezen unieke naam van de regel. Vaak wordt hierbij een zo logisch mogelijke naam gekozen die aanduidt wat de betreffende regel zal doen bij uitvoering. Een voorbeeld van een RegelSpraak-regel hierbij is:

Regel bepaal leeftijd

Syntax Regel specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```
<regel> ::= "Regel" <regelnaam> (\n <regelversie>)+
<regelnaam> ::= <karakterreeks>
```

4.2 Regelversie

Een RegelSpraak-regel bevat na de naam één of meerdere versies van de regel. Iedere regelversie heeft zijn eigen **geldigheidsperiode**, voorafgegaan door het (vaste) woord *geldig*. Hierbij kunnen bij een geldigheidsperiode de volgende intervallen van toepassing zijn:

1. een gesloten interval, bijvoorbeeld *vanaf 01-01-2021 t/m 31-12-2021*;
N.B. een dergelijke interval kan ook gespecificeerd worden door alleen het jaartal te noemen, bijvoorbeeld *vanaf 2020 t/m 2021*;
2. een éézijdig open interval, bijvoorbeeld *vanaf 01-01-2021* of *t/m 31-12-2021*;
3. een tweezijdig open interval, in dat geval staat er het woord "altijd" bij vermeld.

De geldigheidsperiodes van verschillende regelversies van een regel mogen elkaar niet overlappen. Bij het uitvoeren van RegelSpraak-regels wordt de zogenaamde **rekendatum** gebruikt om te bepalen welke regelversie moet worden gebruikt. Voor meer informatie over deze rekendatum wordt verwezen naar paragraaf 5.3.

Een voorbeeld van een RegelSpraak-regel wat betreft een regelversie is als volgt weer te geven:

Regel bepaal leeftijd
geldig altijd

Syntax regelversie specificatie:

```

<regelversie> ::= <versie> \n <regelSpraakregel>

<versie> ::= "geldig" <versiegeldigheid>

<versiegeldigheid> ::= "altijd" | ("vanaf " (<datumwaarde> | <jaar>) ["t/m "
(<datumwaarde> | <jaar>)]) | ("t/m " (<datumwaarde> | <jaar>))

```

4.3 Basispatroon

Tot slot bevat een RegelSpraak-regel een basispatroon bestaande uit een **resultaatdeel**, een **voorwaardendeel** en een **variabelendeel**.

Elke versie van een RegelSpraak-regel begint met een resultaatdeel. Dit resultaatdeel is verplicht. Het beschrijft een **actie** die overeenkomt met datgene wat er gebeurt als de regel wordt toegepast. Er zijn in totaal acht verschillende acties:

- a. Gelijktelling: dit is een actie waarbij een attribuut een waarde krijgt van een expressie, zie verder paragraaf 9.1;
- b. Kenmerktoekenning: dit bepaalt of een instantie van een objecttype een bepaald kenmerk krijgt, zie verder paragraaf 9.2;
- c. ObjectCreatie: hiermee worden nieuwe instanties van objecttypen aangemaakt, zie verder paragraaf 9.3;
- d. FeitCreatie: dit zorgt voor het aanmaken van nieuwe feiten, zie verder paragraaf 9.4;
- e. Consistentieregels: dit zorgt voor het controleren of de waarde van een attribuut, rol of instantie aan bepaalde criteria voldoet, zie verder paragraaf 9.5;
- f. Initialisatie: dit wordt gebruikt om een waarde toe te kennen aan een attribuut waarvan de waarde niet leeg mag zijn, zie verder paragraaf 9.6;
- g. Verdeling: dit wordt gebruikt om een waarde van een numeriek attribuut te verdelen over een attribuut van meerdere instanties van een ander objecttype, zie verder paragraaf 9.7;
- h. Dagsoortdefinitie: dit wordt gebruikt om te definiëren wanneer een datum van een bepaalde dagsoort is, zie verder paragraaf 9.8.

Na het resultaatdeel kan optioneel (met andere woorden niet verplicht) een voorwaardendeel volgen. Het voorwaardendeel beschrijft de voorwaarden waaraan voldaan moet zijn om het resultaatdeel uit te voeren.

Er zijn drie mogelijkheden om het voorwaardendeel uit te drukken:

1. Het voorwaardendeel begint met het woord *indien* met daarna een tijdsafhankelijke enkelvoudige of samengestelde voorwaarde.
2. Het voorwaardendeel begint met *gedurende de tijd dat* gevolgd door een enkelvoudige of samengestelde tijdsafhankelijke voorwaarde.
3. Direct achter het resultaatdeel volgt een tijdsafhankelijke voorwaarde in de vorm van de aanduiding van de periode waarbinnen de regel moet worden toegepast. De periode wordt aangeduid met *vanaf*, *van ... tot*, *van ... tot en met* en */ of tot* (zie paragraaf **Fout! Verwijzingsbron niet gevonden.** voor de betekenis van de periode-aanduidingen).

Een voorwaardendeel kan bestaan uit een enkele elementaire voorwaarde of uit één of meerdere samengestelde voorwaarden. Het voorwaardendeel wordt altijd afgesloten met een punt (.).

N.B. Als één van de condities in een samengestelde voorwaarde tijdsafhankelijk is, dan is de gehele samengestelde voorwaarde tijdsafhankelijk.

Het voorwaardendeel wordt verder uitgewerkt in hoofdstuk 9.9 inclusief diverse voorbeelden.

In het variabelendeel kunnen indien nodig variabelen gespecificeerd worden. Een variabele is een bepaalde waarde of berekening die binnen zowel het resultaatdeel als het voorwaardendeel van de regel, waarin de variabele is gespecificeerd, gebruikt kan worden. Het variabelendeel is het laatste deel van een regel en begint met de woorden *Daarbij geldt*:. Het variabelendeel wordt verder uitgewerkt in hoofdstuk 11 inclusief een voorbeeld.

Syntax basispatroon RegelSpraak:

```
<regel> ::= "Regel" <regelnaam> \n <regelversie> (\n <regelversie>)* \n
<regelSpraakregel>

<regelSpraakregel> ::= <resultaatdeel> \n [<voorwaardendeel>] "." [<variabelendeel>]
```

Tot slot is het van belang te weten in welke **rekenvolgorde** de onderdelen van een basispatroon (resultaat-, voorwaarden- en variabelendeel) afgehandeld worden en hoe deze zich onderling tot elkaar verhouden. Dit zal verder besproken worden in paragraaf 11.1.

4.4 Voorbeeld: regels voor het attribuut leeftijd en de toekenning van het kenmerk *minderjarig*

Om het attribuut *leeftijd* van objecttype *Natuurlijk persoon* binnen de TOKA casus een correcte waarde te geven kunnen we de volgende regel opstellen:

Regel bepaal leeftijd
 geldig altijd
 De *leeftijd* van een *Natuurlijk persoon* moet berekend worden als de tijdsduur van zijn *geboortedatum* tot de *datum van de vlucht* van zijn *reis* in hele jaren.

Merk op dat binnen de TOKA casus vanuit het perspectief van een vlucht wordt geredeneerd. Daarom is hier ook de leeftijd op de datum van de vlucht van belang.

Dit is een eenvoudige regel die geen voorwaardendeel en geen variabelendeel bevat. Een iets ingewikkelder voorbeeld waarbij deze delen wel aanwezig zijn gaat over het kenmerk *minderjarig* van *Natuurlijk persoon*:

Parameter de *volwassenleeftijd* : Numeriek met eenheid jaren

Regel Kenmerktoekenning persoon minderjarig
 geldig altijd
 Een *Natuurlijk persoon* is *minderjarig*
 indien *X* kleiner is dan de *volwassenleeftijd*.
 Daarbij geldt:
X is de tijdsduur van zijn *geboortedatum* tot de *datum van de vlucht* in hele jaren.

Het resultaatdeel van onderstaande regel is een kenmerktoekenning, oftewel een actie die beschrijft dat een instantie het kenmerk krijgt (zie ook paragraaf 9.2):

Een *Natuurlijk persoon* is *minderjarig*

Het voorwaardendeel van de regel is een vergelijking: deze vergelijking beschrijft wanneer dit kenmerk toegekend moet worden. Om dit te bepalen kijken we naar de parameter *volwassenleeftijd* en een variabele *X*:

indien *X* kleiner is dan de *volwassenleeftijd*.

X is tenslotte een variabele die we in het variabelendeel definiëren:

Daarbij geldt:
X is de tijdsduur van zijn *geboortedatum* tot de *datum van de vlucht* in hele jaren.

5 Expressies

In het vorige hoofdstuk is aangegeven dat RegelSpraak-regels opgebouwd zijn volgens een vast patroon met daarin een resultaatdeel en optioneel een voorwaardendeel en variabelendeel. In zowel het resultaatdeel, het voorwaardendeel als het variabelendeel worden **expressies** gebruikt.

Een expressie is een RegelSpraak uitdrukking die één waarde of een lijst van waardes oplevert van een bepaald type. Daarbij kan dat type een datatype zijn (zie paragraaf 3.3) maar ook een objecttype (zie paragraaf 3.1). Een expressie die één waarde oplevert wordt een **enkelvoudige expressie** genoemd en een expressie die een lijst van waardes oplevert wordt een **meervoudige expressie** genoemd. Vaak specificeren we expressies ook met het type dat het oplevert. Een expressie die een enkele numerieke waarde oplevert noemen we bijvoorbeeld een enkelvoudige numerieke expressie.

In een expressie kunnen attributen, parameters of kenmerken gebruikt worden die tijdsafhankelijk zijn. Dit zijn attributen, parameters en kenmerken waarvoor een tijdlijn is gespecificeerd (zie paragraaf 3.8) en die in de tijd verschillende waarden kunnen hebben. Wat dit in algemene zin betekent voor de resultaten van de expressies in vergelijking gebruik van niet-tijdsafhankelijke attributen en parameters is beschreven in paragraaf 5.1.

Er zijn verschillende soorten expressies:

1. Literal expressies
2. Rekendatum en Rekenjaar
3. Tekstreeks expressies
4. Onderwerpexpressies
5. Subselectie expressies
6. Concatenatie expressies
7. Aggregatie expressies
8. Rekenkundige expressies (van zowel numeriek als datum/tijd datatype)
9. Tijdsafhankelijke expressies

In paragraaf 5.2 tot en met 5.8 worden de eerste zeven soorten behandeld.

De “Rekenkundige expressies” worden in hoofdstuk 6 separaat besproken omdat daarmee op velerlei manieren gerekend kan worden.

De specifieke tijdsafhankelijke expressies zijn expressies die alleen kunnen worden toegepast als sprake is van tijdsafhankelijke attributen, kenmerken of parameters (zie paragraaf 3.8). Deze expressies zijn uitgewerkt in hoofdstuk 7.

5.1 Tijdsafhankelijkheid

5.1.1 Tijdsafhankelijk versus niet-tijdsafhankelijk

De meeste expressies kunnen gebruikt worden met zowel tijdsafhankelijke gegevens (zie paragraaf 3.8 Tijdlijnen) als niet-tijdsafhankelijke gegevens.

Als een expressie geen gebruik maakt van tijdsafhankelijke gegevens dan is ook het resultaat van de expressie niet tijdsafhankelijk. De afgeleide waarde heeft dan geen specifieke periode waarin die van toepassing is en kan beschouwd worden als altijd van toepassing.

Als een expressie wel een tijdsafhankelijk element bevat, dan is het resultaat ook tijdsafhankelijk (de enige uitzondering hierop is de specifieke tijdsafhankelijke aggregatie-expressie; zie paragraaf 7.1).

Punten op de tijdlijnen waarop de waarde kan veranderen noemen we knips. De tijdlijn van het attribuut waarvoor de waarde wordt afgeleid moet knips bevatten op alle punten waarop de expressie knips heeft. Als het resultaat van de expressie knips op elke dag dag kan bevatten, dan kan de tijdlijn van het afgeleide attribuut geen tijdlijn voor elke maand zijn, maar moet ook een tijdlijn voor elke dag zijn.

5.1.2 Tijdsafhankelijkheid en eenheidssysteem Tijd

Bij de specificatie van een attribuut, parameter of kenmerk kan worden vastgelegd dat het gegeven tijdsafhankelijk is door een tijdlijn op te nemen in de specificatie. Deze tijdlijn geeft aan op welke momenten op de kalender de waarde van het gegeven kan veranderen (zie paragraaf 3.8). De perioden op een tijdlijn worden uitgedrukt in de eenheden van het eenheidssysteem Tijd

Daarnaast kan bij attributen en parameters met een numeriek datatype ook een eenheid worden gespecificeerd. Daarbij kan ook het eenheidssysteem Tijd (zie paragraaf 3.7) worden gebruikt. Een attribuut kan bijvoorbeeld als datatype een numerieke waarde in € per maand hebben. Vaak zal het logisch zijn dat de tijdseenheid bij het datatype van een attribuut of parameter dezelfde is als de periode van de tijdlijn van het attribuut of de parameter, maar dat is niet noodzakelijk. In de uitgevoerde berekening is er geen verband tussen de tijdlijn en de eventuele tijdseenheid van een gegeven.

De enige expressie waarbij in de berekening het eenheidssysteem Tijd gebruikt wordt in combinatie met de tijdlijn van de waarden van een attribuut of parameter is de expressie “het tijdsevenredig deel” (zie paragraaf 7.3.2).

In alle andere expressies kunnen verschillende combinaties van tijdlijnen en gebruikte tijdseenheden bij datatypes voorkomen, maar in de berekening werken ze onafhankelijk van elkaar. Een voorbeeld daarvan is uitgewerkt in paragraaf 5.1.4.2.

5.1.3 Specificatie van perioden

De waarden van tijdsafhankelijke attributen, parameters of kenmerken hebben een periode waarin ze van toepassing zijn. Die periode kan als volgt worden gespecificeerd:

- Vanaf...
Dit houdt in dat de waarde van toepassing is met ingang van opgegeven jaar, maand of dag.
Voorbeelden:
 - Als een waarde van toepassing is vanaf 2020, dan houdt dat in dat de waarde van toepassing is op 1 januari 2020 en alle dagen daarna .
 - Als een waarde van toepassing is vanaf mei 2020, dan houdt dat in dat de waarde van toepassing is op 1 mei 2020 en alle dagen daarna.
 - Als een waarde van toepassing is vanaf 12 augustus 2020, dan houdt dat in dat de waarde van toepassing is op 12 augustus 2020 en alle dagen daarna.
- Tot en met ...
Dit houdt in dat de waarde van toepassing is in /op het / de volledige opgegeven jaar,

maand of dag.

Voorbeelden:

- Als een waarde van toepassing is tot en met 2020, dan houdt dat in dat de waarde van toepassing is op 31 december 2020 en alle dagen daarvoor .
- Als een waarde van toepassing is tot en met mei 2020, dan houdt dat in dat de waarde van toepassing is op 31 mei 2020 en alle dagen daarvoor.
- Als een waarde van toepassing is tot en met 12 augustus 2020, dan houdt dat in dat de waarde van toepassing is op 12 augustus 2020 en alle dagen daarvoor.

- Tot ...

Dit houdt in dat de waarde niet meer van toepassing is voor het / de opgegeven jaar, maand of dag, maar wel voor alle voorafgaande jaren, maanden of dagen.

Voorbeelden:

- Als een waarde van toepassing is tot 2020, dan houdt dat in dat de waarde van toepassing is op 31 december 2019 en alle dagen daarvoor .
- Als een waarde van toepassing is tot mei 2020, dan houdt dat in dat de waarde van toepassing is op 30 april 2020 en alle dagen daarvoor.
- Als een waarde van toepassing is tot 12 augustus 2020, dan houdt dat in dat de waarde van toepassing is op 11 augustus 2020 en alle dagen daarvoor.

- Van ... tot en met ...

Dit houdt in dat de waarde van toepassing is voor beide opgegeven jaren, maanden of dagen en alle daartussen gelegen jaren, maanden of dagen.

Voorbeelden:

- Als een waarde van toepassing is van 2019 tot en met 2020, dan houdt dat in dat de waarde van toepassing is op 1 januari 2019 en op 31 december 2020 en alle dagen daartussen.
- Als een waarde van toepassing is van mei 2019 tot en met juni 2020, dan houdt dat in dat de waarde van toepassing is op 1 mei 2019 en op 30 juni 2020 en alle dagen daartussen.
- Als een waarde van toepassing is van 19 september 2019 tot en met 12 augustus 2020, dan houdt dat in dat de waarde van toepassing is op 19 september 2019 en op 12 augustus 2020 en alle dagen daartussen.

- Van ... tot ...

Dit houdt in dat de waarde van toepassing is op het / de eerste opgegeven jaar, maand of dag en alle volgende jaren, maanden of dagen, maar niet meer op het / de tweede opgegeven jaar, maand, week of dag.

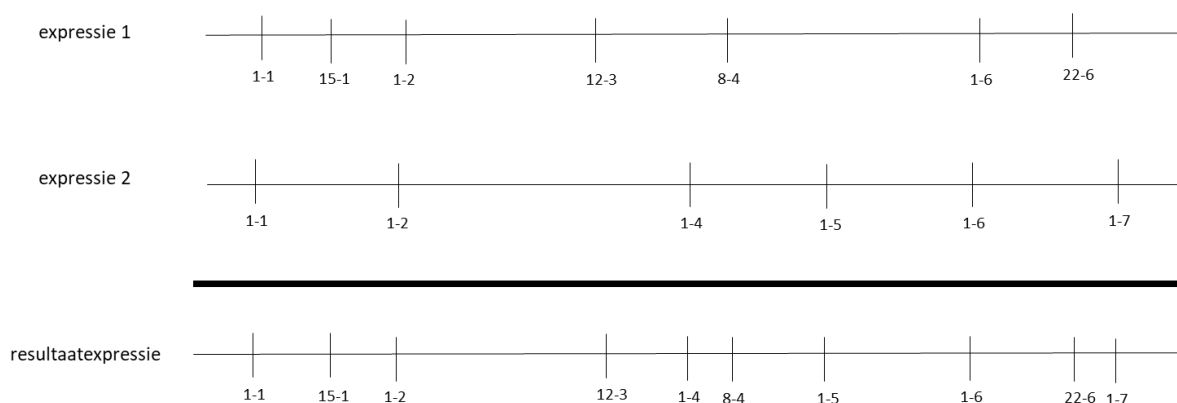
Voorbeelden:

- Als een waarde van toepassing is van 2019 tot 2020, dan houdt dat in dat de waarde van toepassing is op 1 januari 2019 en op 31 december 2019 en alle dagen daartussen.
- Als een waarde van toepassing is van mei 2019 tot juni 2020, dan houdt dat in dat de waarde van toepassing is op 1 mei 2019 en op 31 mei 2020 en alle dagen daartussen.
- Als een waarde van toepassing is van 19 september 2019 tot 12 augustus 2020, dan houdt dat in dat de waarde van toepassing is op 19 september 2019 en op 11 augustus 2020 en alle dagen daartussen.

5.1.4 Resultaat tijdsafhankelijke expressies

Voor het bepalen van het resultaat van een expressie die gebaseerd is op tijdsafhankelijke gegevens wordt de expressie toegepast voor elke periode tussen twee knips op de gecombineerde tijdlijn voor het resultaat.

In onderstaand voorbeeld staat onder de zwarte balk de tijdlijn van de resultaatexpressie met alle knips op basis van de tijdlijnen van de twee expressies die boven de zwarte balk staan, waaruit het resultaat wordt afgeleid.



Hieronder zijn twee voorbeelden uitgewerkt bij toepassing van de plus-expressie. Deze expressie houdt in dat een afgeleide waarde voor een attribuut wordt bepaald door de optelling van de waarden van twee andere gegevens. Deze expressie is verder beschreven in paragraaf 6.2. De regel in het voorbeeld is een Gelijktelling (zie paragraaf 9.1 voor verdere toelichting). Opmerking: Voor perioden waarin geen waarden aanwezig zijn worden deze als leeg beschouwd. In het geval van de plus-expressie wordt een lege waarde als een 0-waarde behandeld.

De regel voor de voorbeelden luidt:

De **te betalen belasting** van een **passagier** moet berekend worden als zijn **belasting op basis van afstand** plus zijn **belasting op basis van reisduur**

Als de attributen niet tijdsafhankelijk zijn (en dus geen tijdlijn hebben waarop waarden kunnen veranderen) dan is het resultaat één waarde voor het attribuut te betalen belasting zonder een specifieke periode waarin die waarde van toepassing is. De waarde kan als altijd van toepassing worden beschouwd.

Als bijvoorbeeld *belasting op basis van afstand* een waarde €50/mnd heeft en *belasting op basis van reisduur* een waarde €20/mnd, dan is het resultaat (niet-tijdsafhankelijk) €70/mnd.

De voorbeelden hieronder illustreren situaties met tijdsafhankelijke gegevens.

5.1.4.1 Voorbeeld 1 - Eenvoudig voorbeeld met gelijke tijdlijnen en eenheden

De attributen *te betalen belasting*, *belasting op basis van afstand* en *belasting op basis van reisduur* hebben allemaal een numeriek datatype (geheel getal) met de eenheid €/maand. Alle attributen hebben ook een tijdlijn “voor elke maand”. Dat houdt in dat de attributen tijdsafhankelijk zijn en dat de waarde per maand kan wijzigen.

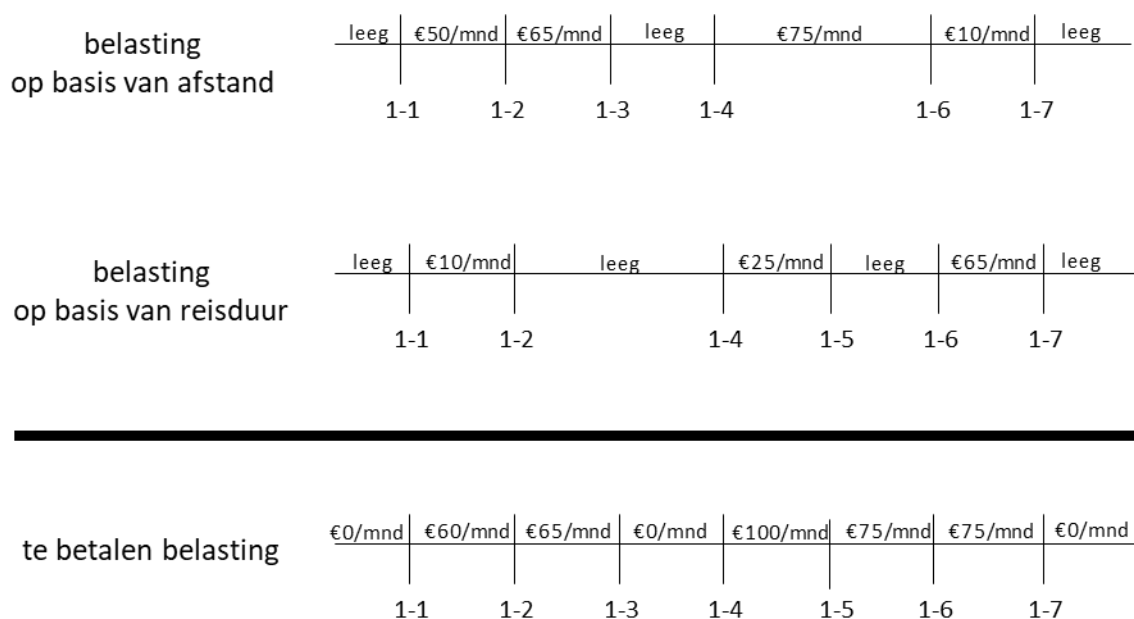
In het voorbeeld hebben de attributen van de plus-expressie de volgende waarden in de tijd:

attribuut	waarde	van	tot
belasting op basis van afstand	€50/mnd	1-1-2024	1-2-2024
	€65/mnd	1-2-2024	1-3-2024
	€75/mnd	1-4-2024	1-6-2024
	€10/mnd	1-6-2024	1-7-2024
belasting op basis van reisduur	€10/mnd	1-1-2024	1-2-2024
	€25/mnd	1-4-2024	1-5-2024
	€65/mnd	1-6-2024	1-7-2024

Voor het bepalen van het resultaat van de optelling worden op de tijdlijn van het attribuut *te betalen belasting* de waarden per maand berekend. Het resultaat wordt:

attribuut	waarde	van / vanaf	tot
te betalen belasting	€0/mnd		1-1-2024
	€60/mnd	1-1-2024	1-2-2024
	€65/mnd	1-2-2024	1-3-2024
	€0/mnd	1-3-2024	1-4-2024
	€100/mnd	1-4-2024	1-5-2024
	€75/mnd	1-5-2024	1-6-2024
	€75/mnd	1-6-2024	1-7-2024
	€0/mnd	1-7-2024	

Grafisch weergegeven ziet het voorbeeld er zo uit:



De waarde van te betalen belasting is voor de perioden van 1-5 tot 1-6 en van 1-6 tot 1-7 gelijk. Omdat de waarde niet verandert op 1-6 bevat de tijdlijn van het attribuut "te betalen belasting" op 1-6 geen knip. Het uiteindelijke resultaat is:

attribuut	waarde	van / vanaf	tot
te betalen belasting	€0/mnd		1-1-2024
	€60/mnd	1-1-2024	1-2-2024
	€65/mnd	1-2-2024	1-3-2024
	€0/mnd	1-3-2024	1-4-2024
	€100/mnd	1-4-2024	1-5-2024
	€75/mnd	1-5-2024	1-7-2024
	€0/mnd	1-7-2024	

belasting
op basis van afstand

leeg	€50/mnd	€65/mnd	leeg	€75/mnd	€10/mnd	leeg
1-1	1-2	1-3	1-4	1-6	1-7	

belasting
op basis van reisduur

leeg	€10/mnd	leeg	€25/mnd	leeg	€65/mnd	leeg
1-1	1-2	1-4	1-5	1-6	1-7	

te betalen belasting	€0/mnd	€60/mnd	€65/mnd	€0/mnd	€100/mnd	€75/mnd	€0/mnd
	1-1	1-2	1-3	1-4	1-5	1-7	

5.1.4.2 Voorbeeld 2 - Complex voorbeeld met verschillende tijdlijnen en eenheden

Ook in dit tweede voorbeeld hebben de attributen *belasting op basis van afstand* en *belasting op basis van reisduur* een numeriek datatype (geheel getal) met de eenheid €/maand. Maar het attribuut *te betalen belasting* heeft een numeriek datatype (geheel getal) met de eenheid €/jaar. Dit betekent dat in de berekening de som van bedragen in € per maand met de omrekenfactor (factor 12) uit het eenheidssysteem Tijd wordt omgerekend naar een bedrag in € per jaar (N.B. Deze omrekening vindt plaats op basis van het eenheidssysteem en wordt ook uitgevoerd als sprake is van niet-tijdsafhankelijke gegevens).

Daarnaast heeft het attribuut *belasting op basis van afstand* een tijdlijn “voor elke dag”. Dit houdt in dat de waarde van dit attribuut op elke dag kan veranderen. Het attribuut *belasting op basis van reisduur* heeft een tijdlijn “voor elke maand” en de waarde kan dus maandelijks wijzigen.

De tijdlijn van het attribuut waarvoor de waarde wordt afgeleid, *te betalen belasting*, moet de aansluiten op de knips van de tijdlijnen van de gegevenselementen uit de expressie. Dat betekent dat voor *te betalen belasting* de tijdlijn “voor elke dag” is.

Gespecificeerd met GegevensSpraak:

belasting op basis van afstand Numeriek (geheelgetal) met eenheid €/mnd voor elke dag;
belasting op basis van reisduur Numeriek (geheelgetal) met eenheid €/mnd voor elke maand;

te betalen belasting Numeriek (geheelgetal) met eenheid €/jr voor elke dag;

In het voorbeeld hebben de attributen van de plus-expressie de volgende waarden in de tijd:

attribuut	waarde	van	tot
belasting op basis van afstand	€50/mnd	1-1-2024	15-1-2024
	€65/mnd	1-2-2024	12-3-2024
	€75/mnd	8-4-2024	1-6-2024
	€10/mnd	1-6-2024	22-6-2024
belasting op basis van reisduur	€10/mnd	1-1-2024	1-2-2024
	€25/mnd	1-4-2024	1-5-2024
	€65/mnd	1-6-2024	1-7-2024

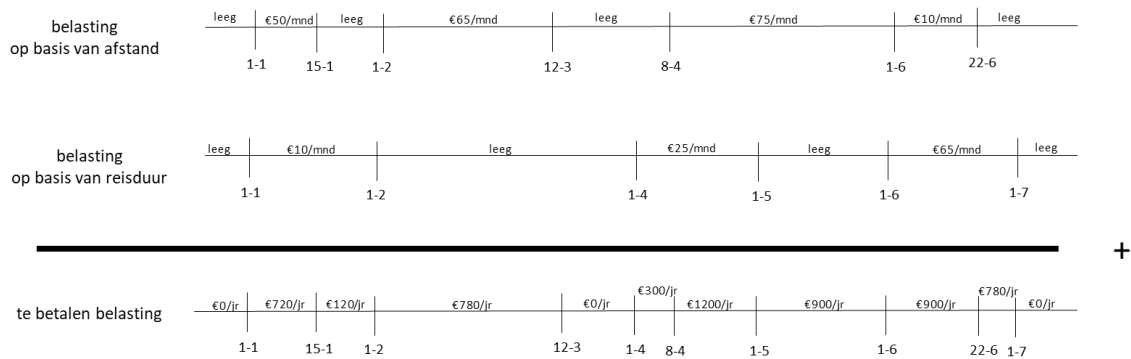
Voor het bepalen van het resultaat van de optelling moeten op de tijdlijn van het attribuut *te betalen belasting* de waarden per jaar worden berekend. Hiervoor worden eerst de perioden afgeleid waarin de waarden wijzigen en vervolgens worden de opgetelde waarden in €/mnd berekend:

attribuut	waarde	van / vanaf	tot
te betalen belasting	€0/mnd		1-1-2024
	€60/mnd	1-1-2024	15-1-2024
	€10/mnd	15-1-2024	1-2-2024
	€65/mnd	1-2-2024	12-3-2024
	€0/mnd	12-3-2024	1-4-2024
	€25/mnd	1-4-2024	8-4-2024
	€100/mnd	8-4-2024	1-5-2024
	€75/mnd	1-5-2024	1-6-2024
	€75/mnd	1-6-2024	22-6-2024
	€65/mnd	22-6-2024	1-7-2024
	€0/mnd	1-7-2024	

Daarna wordt dit omgerekend naar €/jr. Het resultaat wordt:

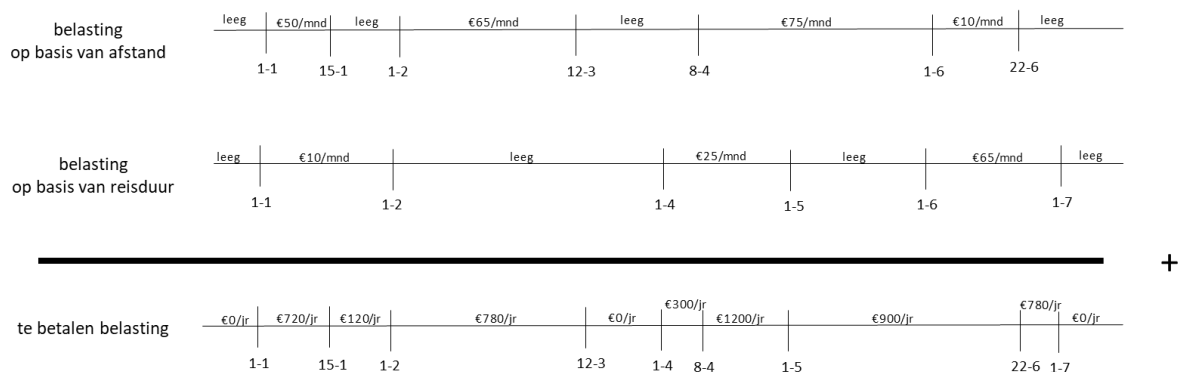
attribuut	waarde	van / vanaf	tot
te betalen belasting	€0/jr		1-1-2024
	€720/jr	1-1-2024	15-1-2024
	€120/jr	15-1-2024	1-2-2024
	€780/jr	1-2-2024	12-3-2024
	€0/jr	12-3-2024	1-4-2024
	€300/jr	1-4-2024	8-4-2024
	€1200/jr	8-4-2024	1-5-2024
	€900/jr	1-5-2024	1-6-2024
	€900/jr	1-6-2024	22-6-2024
	€780/jr	22-6-2024	1-7-2024
	€0/jr	1-7-2024	

Grafisch weergegeven ziet het voorbeeld er zo uit:



De waarde van te betalen belasting is voor de aaneengesloten perioden van 1-5 tot 1-6 en van 1-6 tot 22-6 is gelijk. Omdat de waarde niet verandert op 1-6 bevat de tijdlijn van het attribuut “te betalen belasting” op 1-6 geen knip. Het uiteindelijke resultaat is:

attribuut	waarde	van / vanaf	tot
te betalen belasting	€0/jr		1-1-2024
	€720/jr	1-1-2024	15-1-2024
	€120/jr	15-1-2024	1-2-2024
	€780/jr	1-2-2024	12-3-2024
	€0/jr	12-3-2024	1-4-2024
	€300/jr	1-4-2024	8-4-2024
	€1200/jr	8-4-2024	1-5-2024
	€900/jr	1-5-2024	22-6-2024
	€780/jr	22-6-2024	1-7-2024
	€0/jr	1-7-2024	



5.2 Literal expressies

Een **literal expressie** is een enkelvoudige expressie die bestaat uit een enkele waarde met eventueel, in het geval van een numeriek datatype, een eenheid. Literal expressies worden gebruikt om o.a. een waarde toe te kennen aan een attribuut of parameter, om ermee te rekenen, of om ermee te vergelijken. De volgende vijf expressies zijn voorbeelden van literal expressies:

9

08-12-2022

'Amsterdam Schiphol'

waar

49 jr

N.B: als de waarde van het datatype *Tekst* is, dan moet de waarde zelf tussen dubbele aanhalingstekens ("*<waarde>*") staan. Dit geldt niet voor waarden in een enumeratie (zie paragraaf 3.4.2); deze worden uitgedrukt met enkele aanhalingstekens wanneer ze in een regel aangehaald worden ('*<waarde>*').

Syntax literal expressie specificatie:

```

<literalexpressie> ::= <booleanwaarde> | <getalwaarde> | <dedato> | <enumeratiewaarde> |
<tekstwaarde>

<booleanwaarde> ::= ("waar" | "onwaar")

<getalwaarde> ::= <getal> [(<eenheidsafkorting>)+] | (<eenheidsafkorting>+ "/"
<eenheidsafkorting>+)]

<dedato> ::= "dd. "<datumwaarde> [<tijdwaarde>]
<datumwaarde> ::= <dag>"-"<maand>"-"<jaar>
<tijdwaarde> ::= <uur>":"<minuut>":"<seconde>"."<milliseconde>

<enumeratiewaarde> ::= "'" <karakterreeks> "'"
<tekstwaarde> ::= "\" <karakterreeks> "\"

```

5.3 Rekendatum en Rekenjaar

Bij het uitvoeren van RegelSpraak-regels is het van belang om te weten welke versie van een regel (zie paragraaf 4.2) of welke parameterwaardes gebruikt moeten worden. **Rekendatum** en **Rekenjaar** zijn expressies die daarvoor gebruikt worden. De datum respectievelijk het jaar dat deze twee expressies opleveren moet bij het uitvoeren van RegelSpraak-regels opgegeven worden.

In het onderstaande voorbeeld staan twee varianten om de omzetbelasting op energie te berekenen. De rekendatum bepaalt welke van de varianten uitgevoerd zal worden. Als de rekendatum tussen 1 juli 2022 en 31 december 2022 (inclusief grenzen) ligt, dan zal de middelste regelversie met het lage btw percentage worden gebruikt. Bij andere rekendata zal de eerste of laatste regelversie met het hoge btw percentage worden gebruikt.

Regel bepaal af te dragen omzetbelasting op energie
 geldig t/m 30-06-2022
 De af te dragen omzetbelasting van een Eenheid energie moet berekend worden als het HOGE BTW PERCENTAGE maal de prijs van de Eenheid energie.
 geldig vanaf 01-07-2022 t/m 31-12-2022
 De af te dragen omzetbelasting van een Eenheid energie moet berekend worden als het LAGE BTW PERCENTAGE maal de prijs van de Eenheid energie.
 geldig vanaf 01-01-2023
 De af te dragen omzetbelasting van een Eenheid energie moet berekend worden als het HOGE BTW PERCENTAGE maal de prijs van de Eenheid energie.

De rekendatum kan daarnaast ook als expressie in de regels gebruikt worden. We zouden het bijvoorbeeld kunnen gebruiken in plaats van de *datum van de vlucht* om een *leeftijd* te bepalen:

Regel bepaal leeftijd op rekendatum
 geldig altijd
 De leeftijd van een Natuurlijk persoon moet berekend worden als de tijdsduur van zijn geboortedatum tot de Rekendatum in hele jaren.

5.4 Tekstreeks expressies

In expressies, die een waarde van het datatype *Tekst* opleveren, kan gebruik worden gemaakt van de expressie “reeks van teksten en waarden”. Dit is een expressie die gebruikt wordt om bij uitvoering van de regel de waarden van een expressie om te zetten naar een tekstwaarde, ook als de attributen een ander datatype hebben dan “Tekst”. Attributen in een reeks van teksten en waarden worden in dat geval omringd door “«»” haken.

Bijvoorbeeld: in ons objecttype *Natuurlijk persoon* heeft het attribuut *identificatienummer* het datatype *Numeriek (positief geheel getal)*. Onderstaande expressie is dan ook een numerieke expressie en heeft als resultaat een getal, bijvoorbeeld 528:

Het *identificatienummer* van de *Natuurlijk persoon*

Soms kan het nodig zijn om geen getal maar een karakterreeks te gebruiken. Onderstaande expressie is een Tekst expressie en heeft als resultaat een karakterreeks, bijvoorbeeld “528”:

«het *identificatienummer* van de *Natuurlijk persoon*»

Syntax reeks van teksten en waarden specificatie:

```
<tekstenwaardereeks> ::= "\" ("«" <expressie> "»") | <karakterreeks>)+ "\"
```

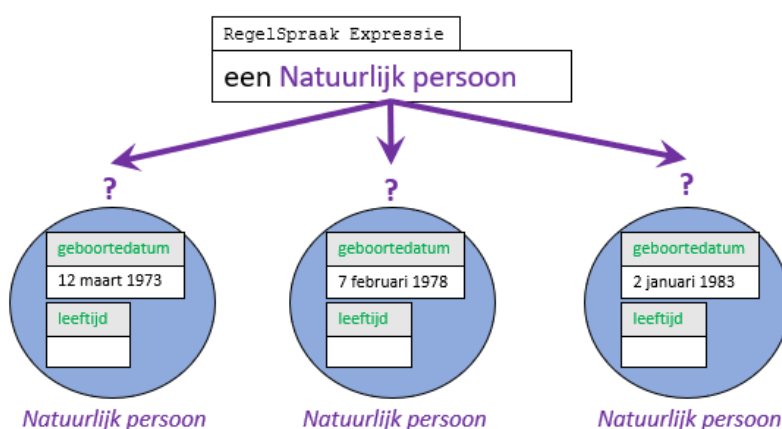
5.5 Onderwerpexpressies

Een volgend soort expressie is de **onderwerpexpressie**. Dit is een expressie die bepaalt waar een regel precies op toegepast moet worden. Een onderwerpexpressie bestaat uit of een verwijzing naar een objecttype, of een attribuut van een objecttype, of een rol die door een objecttype ingevuld wordt. Bij de uitvoering van RegelSpraak wordt deze verwijzing gebonden aan de instanties van het desbetreffende objecttype en wordt de desbetreffende regel uitgevoerd met deze instanties. Elke regel moet minstens één onderwerpexpressie bevatten waarvan tenminste één een zogenaamd universeel onderwerp moet zijn.

5.5.1 Onderwerp

De meest eenvoudige onderwerpexpressie bestaat uit de naam van het bedoelde objecttype.
een *Natuurlijk persoon*

Deze expressie is een verwijzing naar een willekeurige instantie van *Natuurlijk persoon*.



Figuur 10: een eenvoudige onderwerpexpressie die verwijst naar een willekeurige instantie van *Natuurlijk persoon*.

Er kan ook op een andere manier naar zo'n willekeurige instantie worden verwezen dan met de naam van het objecttype. Er kan namelijk gebruik gemaakt worden van een rol of van een niet-bezittelijk en niet-bijvoeglijk kenmerk, zoals te zien in onderstaande voorbeelden:

een **passagier**

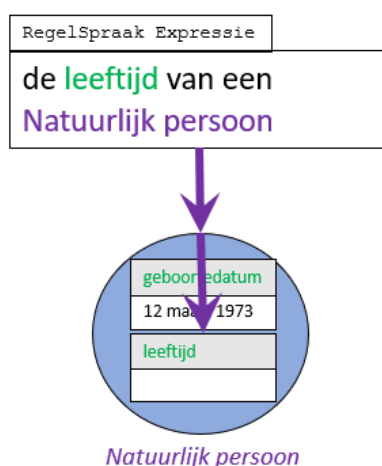
een **belaste reis**

5.5.2 Selectie

De meeste RegelSpraak-regels veranderen de waarden van attributen van instanties van objecttypen. In een onderwerpexpressie kunnen we naar een dergelijk attribuut verwijzen door gebruik te maken van een **selectie**. Met een selectie kan verwezen worden naar een specifiek attribuut of rol van de instanties waarnaar het onderwerp verwijst. Dat specifieke attribuut of rol noemen we een **selector**. Een selectie wordt gespecificeerd door deze selector, gevolgd door het woord *van* en een onderwerpexpressie.

Als we bijvoorbeeld iets willen doen met de *leeftijd* van een *Natuurlijk persoon* dan hebben we een selectie nodig. De selector is hier de combinatie van een lidwoord en de naam van een attribuut, te weten *de leeftijd*:

de **leeftijd** van een **Natuurlijk persoon**



Figuur 11: selectie bij een universeel onderwerp.

5.5.3 Universeel onderwerp

Zoals opgemerkt in paragraaf 3.1 is het in RegelSpraak niet mogelijk om te verwijzen naar een specifieke instantie van een objecttype. Pas bij het uitvoeren van RegelSpraak worden de verwijzingen uit onderwerpexpressies gebonden aan specifieke instanties, en zal de regel opnieuw uitgevoerd worden voor elk van deze instanties. Hoe dit precies in zijn werk gaat zal in deze paragraaf worden toegelicht.

We hebben onderstaande regel om de leeftijd van een *Natuurlijk persoon* te bepalen op basis van zijn *geboortedatum* en de, voor dit voorbeeld willekeurig gekozen, datum 11 november 2022. We maken in deze regel gebruik van de rekenkundige expressie *tijdsduur* die in paragraaf 6.9 verder zal worden uitgelegd.

Regel bepaal leeftijd

geldig altijd

De **leeftijd** van een **Natuurlijk persoon** moet berekend worden als de tijdsduur van de

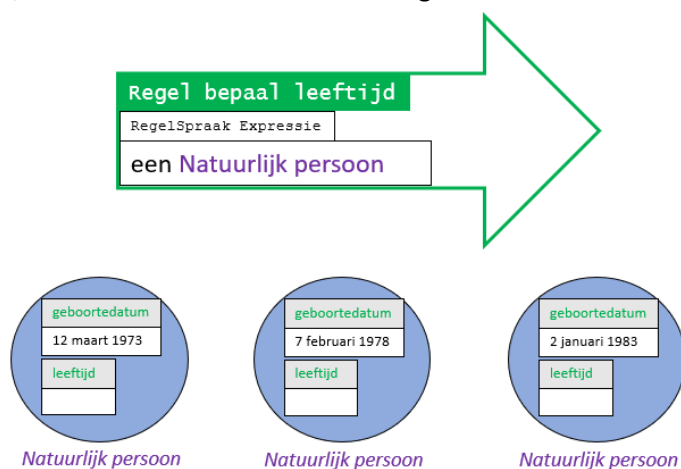
geboortedatum van de *Natuurlijk persoon* tot 11 november 2022 in hele jaren.

N.B.: Deze regel zal in RegelSpraak anders worden weergegeven aangezien het objecttype *bezield* is. Dit zal in de volgende paragraaf worden uitgelegd.

Deze regel bevat twee onderwerpexpressies die allebei naar een ander attribuut van *Natuurlijk persoon* verwijzen. Dit zijn twee verschillende soorten onderwerpexpressies. Een onderwerpexpressie waarin het lidwoord *een* wordt gebruikt noemen we een **universeel onderwerp**. Elke RegelSpraak-regel moet minstens één universeel onderwerp bevatten. Een universeel onderwerp geeft aan op welke instanties de regel toegepast moet worden; zonder een universeel onderwerp kan de regel niet uitgevoerd worden. In onze regel is de eerste onderwerpexpressie (*de leeftijd van een Natuurlijk persoon*) het universele onderwerp. Wat dat betreft zou je een universeel onderwerp kunnen zien als een soort logische formule: ALS er een instantie is die voldoet aan het universele onderwerp, DAN moet de regel uitgevoerd worden met die instantie.

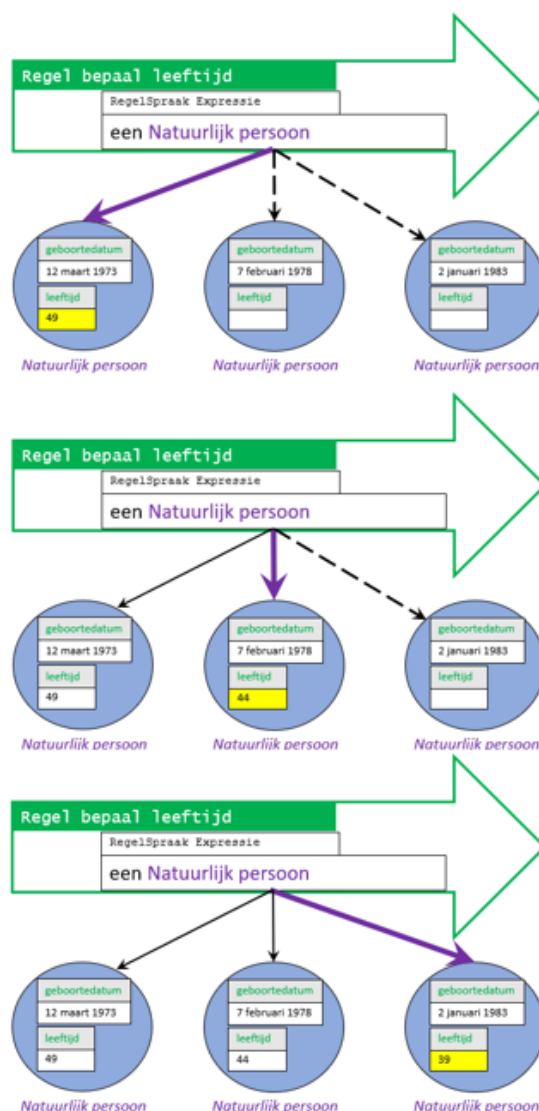
Als we een universeel onderwerp hebben, dan kunnen we het objecttype dat er in gebruikt wordt ook gebruiken in **niet-universele onderwerpexpressies**. Deze andere onderwerpexpressies worden dan met de lidwoorden *de* of *het* geschreven. De tweede onderwerpexpressie (*de geboortedatum van de Natuurlijk persoon*) in onze regel verwijst naar het attribuut *geboortedatum* van dezelfde instantie van *Natuurlijk persoon* als de instantie in het universele onderwerp. Oftewel, naar de instantie waarvan het attribuut *leeftijd* berekend moet worden.

Stel, we hebben drie instanties van *Natuurlijk persoon* en onze regel ten behoeve van het bepalen van de leeftijd, zoals te zien in onderstaande figuur:



Figuur 12: De regel *bepaal leeftijd* en drie instanties van *Natuurlijk persoon*.

Bij het uitvoeren van RegelSpraak zal deze regel drie keer toegepast worden, elke keer op een andere instantie. Immers, het universele onderwerp verwijst naar objecttype *Natuurlijk persoon* en er zijn drie instanties van dit objecttype. Dit kan geïllustreerd worden zoals hierna is weergegeven in Figuur 13. Hier betekent de paarse doorgetrokken pijl dat de berekening in behandeling is, de zwarte gestippelde pijl geeft aan dat de berekening nog uitgevoerd moet gaan worden, en de zwarte doorgetrokken pijl betekent dat de berekening is uitgevoerd.



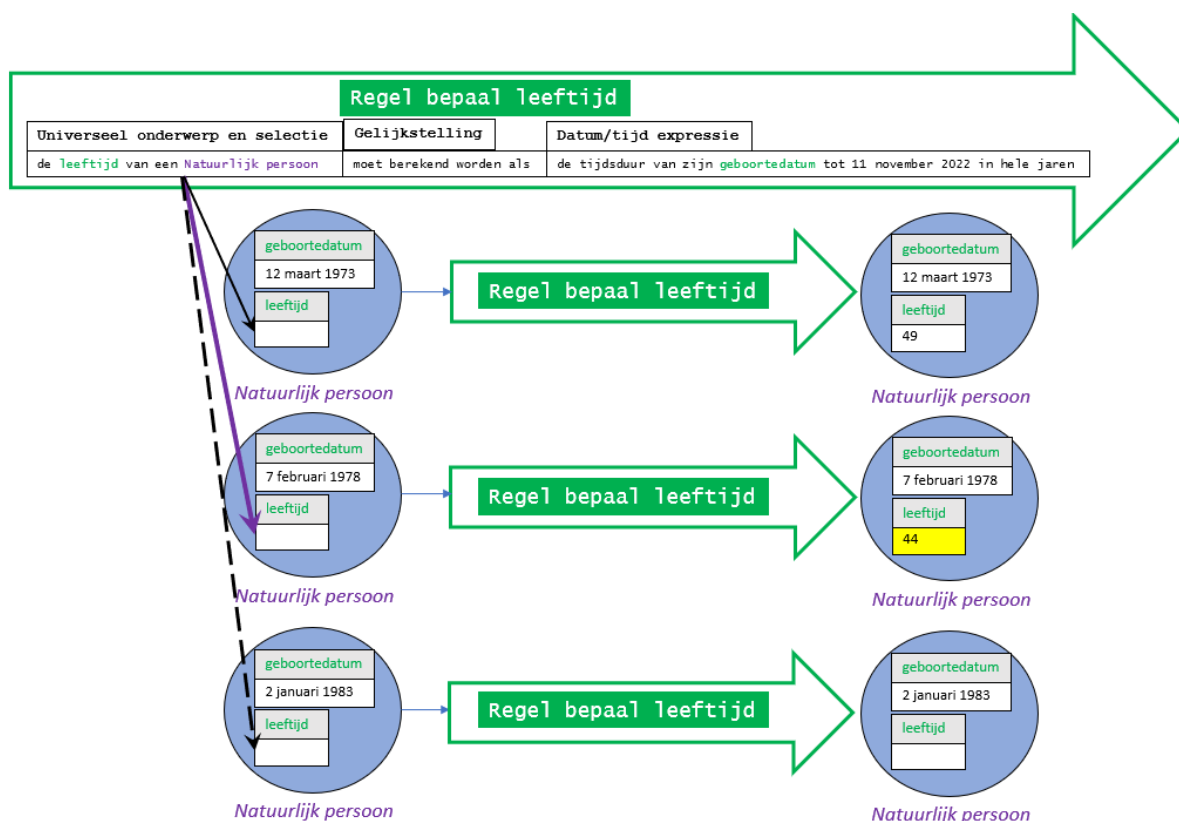
Figuur 13: de regel *bepaal leeftijd* wordt drie keer uitgevoerd, elke keer voor een andere instantie van *Natuurlijk persoon*.

5.5.4 Bezielde objecttypen

Zoals eerder opgemerkt zal onze voorbeeldregel anders weergegeven worden in RegelSpraak. De reden hiervoor is dat het objecttype van het universele onderwerp *Natuurlijk persoon* beziel is (zie paragraaf 5.5.3). Zoals al eerder uitgelegd in paragraaf 3.1.1 kan naar bezielde objecttypen verwezen worden met een verwijzend voornaamwoord 'hij' en een bezittelijk voornaamwoord 'zijn'. Dit kan echter alleen als er slechts één onderwerp is met een beziel objecttype, wat in ons voorbeeld het geval is. We kunnen het dus over *zijn geboortedatum* hebben in plaats van over *de geboortedatum van een Natuurlijk persoon*. Onze voorbeeldregel uit de vorige paragraaf zou er in RegelSpraak dan ook als volgt uit kunnen (moeten) zien:

Regel bepaal leeftijd
 geldig altijd
 De *leeftijd* van een *Natuurlijk persoon* moet berekend worden als de tijdsduur van zijn *geboortedatum* tot 11 november 2022 in hele jaren.

In de navolgende figuur staat het voorbeeld verder uitgewerkt en geïllustreerd. Ook hier betekent de paarse doorgetrokken pijl dat de berekening in behandeling is, de zwarte gestippelde pijl geeft aan dat de berekening nog uitgevoerd moet gaan worden, en de zwarte doorgetrokken pijl betekent dat de berekening is uitgevoerd.



Figuur 14: de regel *bepaal leeftijd* met het universele onderwerp *een Natuurlijk persoon* wordt voor de tweede keer uitgevoerd (vandaar aan de linkerzijde geen zwarte stippellijn maar een parse doorgetrokken lijn).

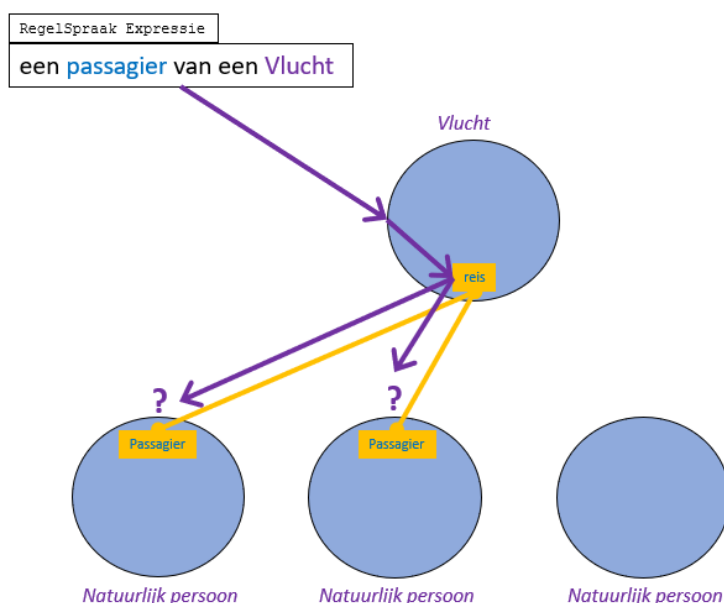
5.5.5 Onderwerpketen

Een onderwerpexpressie kan ook met een rol (zie paragraaf 3.11) worden gespecificeerd. Rollen in een feittype koppelen verschillende objecttypen aan elkaar. Een instantie van een objecttype dat een rol vervult is, door middel van het feittype waar de rol in gespecificeerd is, verbonden met een instantie van een ander objecttype. Als we een rol gebruiken, dan maken we eigenlijk een keten van objecttypen die met feittypen verbonden zijn. Een dergelijke keten noemen we de **onderwerpketen**.

We kunnen bijvoorbeeld een regel maken die uitgevoerd moet worden op elke instantie van *Natuurlijk persoon* die de rol *passagier* invult bij een instantie van *Vlucht*:

een *passagier* van een *Vlucht*

Dit wordt geïllustreerd in de navolgende figuur.



Figuur 15: een onderwerpketen. De verwijzing gaat van de instantie van *Vlucht*, via de rollen *reis* en *passagier* uit het feittype *Vlucht van Natuurlijke personen* naar de relevante instanties van *Natuurlijk persoon*.

Hierbij is het van belang of een rol enkelvoudig of meervoudig is. De rol *passagier* bijvoorbeeld is meervoudig. Eén *Vlucht* betreft de verplaatsing van meerdere passagiers. In dit geval moet het onbepaald lidwoord *een* gebruikt worden. De regel zal dan uitgevoerd worden op elke instantie van *Natuurlijk persoon* die de rol van passagier vervult bij elke instantie van *Vlucht*. De rol *reis* is daarentegen enkelvoudig. In dit geval moet het bepaald lidwoord “de” of “het” gebruikt worden. Een regel met de onderstaande expressie zal uitgevoerd worden op de instantie van *Vlucht* die de rol *reis* invult voor elke instantie van *Natuurlijk persoon*.

De *reis* van een *Natuurlijk persoon*

In het eerste voorbeeld zal voor elk *Natuurlijk persoon* dat de rol *passagier* vervult de regel worden uitgevoerd. Soms is dat echter niet de bedoeling; als we bijvoorbeeld de gemiddelde leeftijd van de *passagiers* op een *Vlucht* willen bepalen, dan moeten we een regel maken die per *Vlucht*-instantie één keer wordt toegepast, op de leeftijden van alle *passagiers* van die *Vlucht*. Dit doen we door gebruik te maken van een **kwantificatie**. We gebruiken in plaats van het woord *zijn* of een bepaald lidwoord een kwantificerend woord:

de *leeftijden* van alle *passagiers* van de *vlucht*

Bij dit soort onderwerpen kunnen meerdere soorten kwantificaties gebruikt worden. In de bovenstaande voorbeelden werd de kwantificatie “alle” gebruikt maar in RegelSpraak mag ook “geen”, “ten minste <getal>”, “hoogstens <getal>” en “precies <getal>” gebruikt worden waarin <getal> vervangen wordt door de tekstuele vorm van de getallen “één”, “twee”, “drie” of “vier”, of meer dan vier (maar dan genoteerd met een getal zoals 6 of 12).

Syntax selectie specificatie:

```
<attribuutvanonderwerp> ::= <attribuutmetlidwoord> "van" [<kwantificatie>]
<onderwerpketen>

<onderwerpketen> ::= ((<lidwoord> | "zijn") (<objecttypenaam> | <rolnaam> |
<kenmerknaam>)) | ((<selector> "van" <onderwerpketen>) | <subselectie>)

<selector> ::= [<lidwoord>] <rolnaam>
```

5.6 Subselectie expressies

Subselectie is het mechanisme dat gebruikt wordt om een groep van instanties of waarden te maken die aan bepaalde selectiecriteria voldoen. Een subselectie bestaat uit een onderwerpexpressie, gevolgd door de woorden *die* of *dat*, en een predicaat. Het predicaat omschrijft de selectiecriteria die gebruikt moeten worden, zie verder hoofdstuk 7.

We hebben bijvoorbeeld eerder al gezien dat we instanties van het objecttype *Natuurlijk persoon* kunnen onderverdelen in minder- en meerderjarig met het kenmerk *minderjarig*. Om deze onderverdeling in een regel te gebruiken passen we subselectie toe waarbij het predicaat een enkel selectie criterium bevat dat controleert op het kenmerk *minderjarig*:

Het aantal minderjarige passagiers van een reis moet berekend worden als het aantal passagiers van de reis die een minderjarige zijn.

Attribuutwaarden kunnen ook gebruikt worden in een predicaat. Het objecttype *Natuurlijke persoon* bevat bijvoorbeeld ook een attribuut *leeftijd*. We zouden bovenstaande regel ook kunnen schrijven met *leeftijd* als selectie criterium:

Het aantal minderjarige passagiers van een reis moet berekend worden als het aantal passagiers van de reis die aan de volgende voorwaarde voldoet:
zijn leeftijd is kleiner dan de volwassenleeftijd.

Een predicaat kan ook meerdere selectiecriteria bevatten. Indien dit het geval is, moeten de woorden *aan alle/geen/precies/ten minste/ten hoogste/van de volgende voorwaarden voldoen* achter het woord *die* staan. Selectiecriteria in een predicaat kunnen een controle van een kenmerk, attribuut of rol zijn, maar kunnen ook andere voorwaarden bevatten. Dit is in basis dezelfde constructie die voor samengestelde voorwaarden wordt gebruikt; voor meer informatie hierover en bijbehorende voorbeelden wordt verwezen naar paragraaf 10.2.

Syntax subselectie specificatie:

```
<subselectie> ::= <onderwerpketen> ("die" | "dat") <predicaat>
<predicaat> ::= <elementairpredicaat> | <samengesteldpredicaat>

<elementairpredicaat> ::= <getalpredicaat> | <tekstpredicaat> | <datumpredicaat> |
<objectpredicaat>

<samengesteldpredicaat> ::= "aan" <kwantificatie> "volgende voorwaarde"["n"] ("voldoet"
| voldoen):" (<samengesteldevoorwaardeonderdeel> | <toplevelvoorwaardevergelijking>)

<getalpredicaat> ::= <topleveltweezijdigegetalvergelijkingoperatormeervoud>
<getalexpressie>

<tekstpredicaat> ::= <topleveltweezijdigetekstvergelijkingoperatormeervoud>
<tekstexpressie>

<datumpredicaat> ::= <topleveltweezijdigedatumvergelijkingoperatormeervoud>
<datumexpressie>

<objectpredicaat> ::= <topleveltweezijdigeobjectvergelijkingoperatormeervoud>
<objectexpressie>
```

5.7 Concatenatie expressie

Het is mogelijk om de verzamelingen instanties of waarden die de selecties opleveren samen te voegen tot een nieuwe verzameling instanties of waarden. Dit noemen we **concatenatie** en wordt in RegelSpraak gedaan door het plaatsen van het woord *en* tussen de selecties indien er exact twee selecties zijn opgenomen in de regel, en anders tussen de twee laatste selecties indien er meer dan twee selecties zijn opgenomen in de regel waarbij de overige voorafgaande selecties worden gescheiden middels een komma.

```
"<expr1> en <expr2>"
"<expr1>, <expr2>, ... en <exprn>"
```

Stel dat we geïnteresseerd zijn in passagiers van een vlucht die hetzij minderjarig zijn, hetzij tussen de 18 en 24 jaar oud zijn, hetzij ouder dan 65 jaar zijn, en het aantal passagiers willen weten dat in deze verzameling zit. We kunnen dan een attribuut *aantal uitzonderingspassagiers* maken en deze de correcte waarde geven door gebruik te maken van een concatenatie:

Het *aantal uitzonderingspassagiers* van een *reis* moet berekend worden als het aantal *passagiers* van de *reis* die een *minderjarige* zijn, *passagiers* van de *reis* die een *passagier van 18 tot en met 24 jaar* zijn en *passagiers* van de *reis* die een *passagier van 65 jaar of ouder* zijn.

Er is één uitzondering op deze manier van formuleren. Als een concatenatie gebruikt wordt in een conditie met het predicaat "is gelijk aan", dan wordt niet "en" maar "of" gebruikt tussen de laatste twee expressies van de reeks (zie hoofdstuk 7 voor beschrijving condities en predicaten).

```
"<expr1> of <expr2>"
"<expr1>, <expr2>, ... of <exprn>"
```

Een voorbeeld hiervan is de vergelijking

- de *luchthaven van vertrek* van de *vlucht* is gelijk aan 'Amsterdam Schiphol' of 'Groningen Eelde'

Syntax concatenatie specificatie:

Opmerking: " of " wordt alleen gebruikt bij een concatenatie in een conditie met het predicaat "is gelijk aan"

```
<concatenatie> ::= <expressie> [ (" " <expressie>)* (" en " | " of ") <expressie>
```

5.8 Aggregatie expressies

Aggregatie expressies zijn expressies die met een meervoudige invoer een enkelvoudig resultaat opleveren. De volgende soorten expressies kunnen meervoudig zijn:

1. Een concatenatie is altijd meervoudig, zie verder paragraaf 5.7.
2. Een selectie kan meervoudig zijn (zie ook paragraaf 5.5.2)
 - a. als het object van de selectie (de onderwerpexpressie achter "van") meervoudig is, of
 - b. als het een selectie met rolselector is en de rol is meervoudig.
3. Een subselectie is altijd meervoudig, zie verder paragraaf 5.6.

In RegelSpraak bestaat een aantal vormen van aggregatie. Deze worden in de subparagrafen hierna verder beschreven, voorafgegaan door de syntax van deze aggregatieopties:

1. telling van meervoudige instanties (zie paragraaf 5.8.1);
2. sommatie van attribuutwaarden van meervoudige instanties (zie paragraaf 5.8.2);
3. minimale/maximale waarde bij meervoudige instanties (zie paragraaf 5.8.3);
4. eerste/laatste waarde bij meervoudige instanties (zie paragraaf 5.8.4);

5. aggregatie over dimensies (zie paragraaf 5.8.5).

Syntax aggregatieexpressies specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```

<aggregatie> ::= <getalaggregatie> | <datumaggregatie> | <dimensieaggregatie>

<getalaggregatie> ::= <getalaggregatiefunctie> <expressie>

<getalaggregatiefunctie> ::= "het aantal" | "de maximale waarde van" | "de minimale
waarde van" | "de som van"

<datumaggregatie> ::= <datumaggregatiefunctie> <expressie>

<datumaggregatiefunctie> ::= "de eerste van" | "de laatste van"

<dimensieaggregatie> ::= (<getalaggregatiefunctie> | <datumaggregatiefunctie>)
<attribuutvanonderwerp> <dimensieselectie>

<dimensieselectie> ::= "over" (<aggregerenoveralldimensies> |
<aggregerenoververzameling> | <aggregerenoverbereik>)

<aggregerenoveralldimensies> ::= "alle" <dimensienaammeervoud>

<aggregerenoververzameling> ::= "de" <dimensienaammeervoud> "vanaf" <> "t/m" <>

<aggregerenoverbereik> ::= "de" <dimensienaammeervoud> "in {" <dimensiewaarde> [{" ", "
<dimensiewaarde>)* "en" <dimensiewaarde>] "}"

```

Syntax overige expressies specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```

<expressie> ::= <getalexpressie> | <objectexpressie> | <datumexpressie> |
<tekstexpressie> | <booleanexpressie> | <expressietussenhaakjes> |
<parametermetlidwoord> | <variabelenaam> | <concatenatie>

<getalexpressie> ::= <begrenzingsexpressie> | <afrondingsexpressie> | <getalfunctie> |
<getalaggregatie> | <attribuutvanonderwerp> | <parametermetlidwoord> | <variabelenaam> |
<getalwaarde> | <rekenjaar> | <jaaruitfunctie> | <maanduitfunctie> | <daguitfunctie>

<objectexpressie> ::= [<kwantificatie>] <onderwerpketen>

<datumexpressie> ::= <datumfunctie> | <attribuutvanonderwerp> | <parametermetlidwoord> |
<variabelenaam> | <dedato> | <datumaggregatie>

<tekstexpressie> ::= <tekstenwaardereeks> | <tekstwaarde> | <attribuutvanonderwerp> |
<parametermetlidwoord> | <variabelenaam>

<booleanexpressie> ::= <booleanwaarde> | <attribuutvanonderwerp> |
<parametermetlidwoord> | <variabelenaam>

```

5.8.1 Telling van instanties: aantal

Het aantal instanties in een verzameling kan bepaald worden met de getalaggregatiefunctie *het aantal*. We kunnen bijvoorbeeld met *het aantal* bepalen hoeveel passagiers een vlucht heeft. Als invoer gebruiken we de meervoudige expressie *passagiers van de reis*, en het resultaat is het aantal instanties van *Natuurlijk persoon* dat de rol *passagier* invult bij de instantie van *Vlucht*. In RegelSpraak ziet dit er als volgt uit:

Rege1 Hoeveelheid passagiers van een reis

geldig altijd

De **hoeveelheid passagiers** van een **reis** moet berekend worden als het aantal **passagiers** van de **reis**.

5.8.2 Sommatie: som van

Een andere vorm van een getalaggregatiefunctie is *de som van* die een **sommatie** van meerdere waardes uitvoert. Bij deze getalaggregatie worden de waardes van een meervoudige numerieke expressie bij elkaar opgeteld. De numerieke expressie kan een verwijzing naar een numeriek attribuut, literal of parameter zijn, maar het kan ook een rekenkundige expressie zijn (zie hoofdstuk 6). Het aantal waarden dat bij een sommatie kan worden opgeteld is onbeperkt, in tegenstelling tot de rekenkundige expressie *plus* (zie paragraaf 6.1.4), waarbij slechts twee waarden kunnen worden opgeteld.

Als de sommatie lege waarden bevat, dan zullen die waarden niet worden opgeteld bij de andere waarden. Als alle attribuutwaarden in de sommatie leeg zijn dan zal het resultaat van de sommatie ook leeg zijn aangezien een lege waarde plus een lege waarde nog steeds een lege waarde is.

N.B. Dit geldt alleen voor sommatie, voor een plus-expressie gelden andere regels voor omgang met leeg waarden. Zie hiervoor paragraaf 6.1.4.

Zoals hiervoor aangegeven geldt de regel ‘als alle waarden in de sommatie leeg zijn, dan zal het resultaat van de sommatie ook leeg zijn’ alleen voor de waarde van de attributen in de expressie: als er literals of parameters in de expressie staan die niet leeg zijn maar de attribuutwaarden in dezelfde expressie zijn wel allemaal leeg, dan zal de sommatie alsnog een lege waarde als resultaat geven, oftewel de waardes van de literals of parameters worden dan niet opgeteld¹⁵.

Hier kan overigens van afgeweken worden door een lege waarde van het resultaat te beschouwen als de waarde 0. Dit moet bij de sommatie expliciet gespecificeerd worden door het toevoegen van “*of 0 als die er niet zijn*”¹⁶. Het resultaat van de sommatie zal dan nooit een lege waarde zijn, maar het getal 0, zelfs als alle attribuutwaarden van de sommatie leeg zijn.

Als voorbeeld kan de *totaal te betalen belasting* van een *Vlucht* bepaald worden door gebruik te maken van sommatie:

Regel Totaal te betalen belasting

geldig altijd

De **totaal te betalen belasting** van een **reis** moet berekend worden als de som van de **te betalen belasting** van alle **passagiers** van de **reis**.

Als afgedwongen moet worden dat *totaal te betalen belasting* geen leegwaarde krijgt, dan kan dat gerealiseerd worden door de toevoeging “*of 0 als die er niet zijn*”. Bovenstaand voorbeeld komt er dan als volgt uit te zien:

Regel Totaal te betalen belasting, controleer voor leegwaarde

geldig altijd

De **totaal te betalen belasting** van een **reis** moet berekend worden als de som van de **te betalen belasting** van alle **passagiers** van de **reis** of 0 als die er niet zijn.

5.8.3 Minimale/maximale waarde van

Via een getalaggregatiefunctie kan ook de minimale of maximale waarde worden bepaald bij

¹⁵ Dit is gedaan om te zorgen dat als er geen invoer is, er ook geen resultaten zullen worden berekend.

¹⁶ Strikt genomen zou achter de waarde 0 nog de van toepassing zijnde eenheid vermeld moeten worden zoals bijvoorbeeld €. Op dit moment is dit niet het geval.

meervoudige expressies. We kunnen bijvoorbeeld de hoogste leeftijd van een *Natuurlijk persoon* op een *Vlucht* bepalen door gebruik te maken van de getalaggregatiefunctie *de maximale waarde van*¹⁷. Als invoer gebruiken we de numerieke expressie *de leeftijd van de passagiers van de reis*: dit levert een verzameling op die bestaat uit de waarden van het attribuut *leeftijd* van alle instanties van *Natuurlijk persoon* die de rol *reis* invullen bij de instantie van *Vlucht*. In RegelSpraak ziet dat er als volgt uit:

Regel Leeftijd oudste passagier
 geldig altijd
 De *leeftijd van de oudste passagier* van een *reis* moet gesteld worden op de maximale waarde van de *leeftijden* van alle *passagiers* van de *reis*.

Voor het berekenen van de minimale of maximale waarden worden lege waarden niet meegeteld. Wanneer alle waarden leeg zijn, wordt het resultaat ook leeg.

5.8.4 Eerste/laatste waarde: de eerste/laatste van

Met de datumaggregatiefuncties *de eerste van* of *de laatste van* kan binnen een verzameling van datumwaarden de eerste (oudste) of laatste (jongste) waarde bepaald worden.

Een voorbeeld betreft het bepalen van de datum-tijd voor het berekenen van de belasting op basis van afstand, rekening houdend met de verwachte en daadwerkelijke datum-tijd van vertrek en gebruikmakend van *de eerste van* expressie:

Regel Datum-tijd voor het berekenen van de belasting op basis van afstand
 geldig altijd
 De *datum-tijd voor het berekenen van de belasting op basis van afstand* van een *vlucht* moet berekend worden als de eerste van de *verwachte datum-tijd van vertrek* van de *vlucht* en de *daadwerkelijke datum-tijd van vertrek* van de *vlucht*.

Een ander voorbeeld betreft het bepalen van het bevestigingstijdstip van een vlucht¹⁸, rekening houdend met enerzijds een berekening en anderzijds een constante waarde en gebruikmakend van *de laatste van* datumaggregatiefunctie:

Parameter de *bevestigingsinterval* : Datum en tijd in millisecondes

Parameter de *eerste boekingsdatum* : Datum in dagen

Regel Bevestigingstijdstip vlucht
 geldig altijd
 Het *bevestigingstijdstip* van een *vlucht* moet berekend worden als de laatste van *A* en *B*.
 Daarbij geldt:
A is het *uiterste boekingstijdstip* van de *vlucht* plus *bevestigingsinterval*
B is *eerste boekingsdatum*.

Wanneer een datumexpressie in de verzameling leeg is, wordt deze datumexpressie niet meegenomen in de berekening voor de eerste of laatste datum. Wanneer alle datumexpressies in de verzameling leeg zijn, wordt het resultaat ook op leeg gezet.

Tot slot: wanneer gebruikt gemaakt wordt van *de eerste van* of *de laatste van* datumaggregatiefunctie moet rekening gehouden worden met de precisie van de datum-tijd expressies die gebruikt worden. Deze moeten namelijk allemaal gelijk zijn.

¹⁷ *de leeftijd van de oudste passagier* is geen attribuut van het objecttype *Vlucht*, maar het wordt hier gebruikt om aggregatie te illustreren.

¹⁸ Deze datum is puur ter illustratie in deze paragraaf benoemd als attribuut van het objecttype de *Vlucht*.

Syntax de eerste/laatste van specificatie:

```

<eerstevan> ::= "de eerste van" <datumexpressie> (" , " <datumexpressie>)* "en"
<datumexpressie>

<laatstevan> ::= "de laatste van" <datumexpressie> (" , " <datumexpressie>)* "en"
<datumexpressie>

```

5.8.5 Aggregatie over dimensies

Aggregatiefuncties kunnen tot slot toegepast worden op gedimensioneerde attributen (zie paragraaf 3.6). De labels die bij een dimensie gebruikt worden, worden namelijk gespecificeerd in een volgorde waarna deze volgorde kan worden gebruikt om een interval te specificeren in een dimensie-expressie. Dit kan door gebruik te maken van *vanaf* en *t/m*: het resultaat van deze dimensie-expressie is de verzameling waarden van het attribuut die bij de labels in het interval horen.

Stel dat we bijvoorbeeld ons *Natuurlijk persoon* objecttype hebben uitgebreid met attributen voor zijn elk jaar betaalde belasting. Een van die attributen hebben we een dimensie gegeven om onderscheid te maken tussen de verschillende jaren dat de persoon belasting heeft betaald.

Objecttype de *Natuurlijk persoon* (mv: Natuurlijke personen) (bezield)
 de betaalde belasting over de afgelopen vier jaar *Bedrag*;
 de betaalde belasting in jaar *Bedrag*, gedimensioneerd met *jaardimensie*

Dimensie de *jaardimensie*, bestaande uit de *jaardimensies* (na het attribuut met voorzetsel van):

1. zes jaar geleden
2. vijf jaar geleden
3. vier jaar geleden
4. drie jaar geleden
5. twee jaar geleden
6. een jaar geleden
7. heden

We kunnen nu een regel schrijven die de *betaalde belasting over de afgelopen vier jaar* bepaalt door een getalaggregatie over de *jaardimensie* te gebruiken. Dit kan er dan als volgt uit zien:

Regel Totaal betaalde belasting over 4 jaar
 geldig altijd

De *betaalde belasting over de afgelopen vier jaar* van een *Natuurlijk persoon* moet berekend worden als de som van zijn *betaalde belasting in jaar* vanaf vier jaar geleden t/m een jaar geleden.

6 Rekenkundige expressies

Numerieke, percentage of datum-tijd expressies kunnen berekeningen met diverse rekenkundige operatoren bevatten. Welke dit allemaal kunnen zijn, wordt beschreven in dit hoofdstuk, voorafgegaan door een beschrijving van de hierbij van toepassing zijnde rekenvolgorde, hoe omgegaan moet worden met afrondingen en de mogelijkheden voor begrenzing van het resultaat.

6.1 Rekenen in RegelSpraak

6.1.1 Rekenvolgorde

Het is mogelijk om meerdere berekeningen in een expressie te gebruiken. De uitkomst van de ene berekening wordt dan gebruikt door de andere berekening. Het is hierbij van belang in welke volgorde de berekeningen worden uitgevoerd. Zo kan bijvoorbeeld “2 maal 4 min 2” (wiskundig: $2 \times 4 - 2$) in principe twee verschillende uitkomsten hebben:

1. als de eerste expressie (2×4) als eerste wordt uitgerekend, dan is de uitkomst 6;
2. als de tweede expressie ($4 - 2$) het eerst wordt uitgerekend, dan is de uitkomst 4.

Om deze onduidelijkheid te voorkomen wordt in RegelSpraak de volgende volgorde aangehouden¹⁹:

1. eerst worteltrekken (expressie “wortel van”),
2. dan vermenigvuldigen (expressie “maal”) en delen (expressies “gedeeld door” en “gedeeld door (ABS)”) met als kanttekening dat bij gelijke prioriteit eerst de linker operatie wordt uitgevoerd,
3. en tenslotte optellen (expressie “plus”) en aftrekken (expressies “min” en “verminderd met”), wederom met als kanttekening dat bij gelijke prioriteit eerst de linker operatie wordt uitgevoerd.

Tevens kan gebruik worden gemaakt van haakjes “(” en “)” om daarmee zelfsturend te zijn wat betreft de operaties die als eerste uitgerekend moeten worden. Als een operatie namelijk tussen haakjes staat, dan zal deze als eerste worden uitgerekend, wederom met als kanttekening dat bij gelijke prioriteit eerst de linker operatie wordt uitgevoerd.

Resumé ten aanzien van het eerder genoemde voorbeeld:

- “2 maal 4 min 2” zal als uitkomst 6 hebben, rekening houdend met bovenstaande uitleg (de expressie “maal” wordt standaard eerst uitgevoerd ten opzichte van de expressie “min”);
- “(2 maal 4) min 2” zal ook als uitkomst 6 hebben, aangezien de expressie “maal” ook in dit geval eerst wordt uitgevoerd als gevolg van het haakjesgebruik;
- “2 maal (4 min 2)” zal als uitkomst 4 hebben, omdat eerst de operatie “4 min 2” wordt uitgerekend als gevolg van het haakjesgebruik.

Verder zal bij het opstellen van RegelSpraak-regels rekening gehouden moeten worden met de volgorde waarin variabelen uit het variabelendeel worden uitgerekend. Voor meer informatie hierover wordt verwezen naar paragraaf 11.1.

¹⁹ Hierbij dient opgemerkt te worden dat het machtsverheffen niet voorkomt in RegelSpraak en derhalve ook niet in deze beschrijving van de toegepaste rekenvolgorde in RegelSpraak.

6.1.2 Getal-representatie

Voor rekenkundige expressies met waarden met een *Numeriek* of *Percentage* datatype kan het aantal decimalen (de precisie) van belang zijn:

- 1) Zoals besproken in paragraaf 3.3.1 kent het Numeriek datatype een verplichte beperking: er moet gespecificeerd worden met wat voor soort getal we te maken hebben, namelijk ofwel een getal met een gespecificeerd aantal decimalen, ofwel een geheel getal (dat per definitie nul decimalen heeft), ofwel een getal met een in basis oneindig aantal decimalen²⁰.
- 2) Dit geldt in basis ook voor het Percentage datatype (zie paragraaf 3.3.5): deze wordt gezien als een gespecialiseerd Numeriek datatype met o.a. dezelfde verplichte beperking om aan te geven met wat voor soort getal we te maken hebben (zie hierboven). Echter, voor het bepalen van het aantal decimalen wordt een percentage ook altijd eerst omgezet naar een getal met decimalen (21% wordt bijvoorbeeld omgezet naar het getal 0,21 en 23,45% wordt omgezet naar het getal 0,2345) waarna de regels voor vermenigvuldiging gelden voor het bepalen van het aantal decimalen.

Het aantal decimalen is dus van belang bij (de input van) berekeningen, ook ten aanzien van het toekennen van het resultaat van een berekening. Als dit rekenresultaat bijvoorbeeld toegekend moet worden aan een attribuut, dan kan dat alleen als het aantal decimalen van het rekenresultaat niet groter is dan het aantal decimalen waarmee het betreffende attribuut is gedefinieerd. Als dit wel het geval is, dan zal het resultaat eerst expliciet afgerond moeten worden, zie hiervoor verder paragraaf 6.1.3.

Numerieke waarden in RegelSpraak worden verder verondersteld allemaal dezelfde precisie te hebben. In wetenschappelijke of technische contexten wordt er echter vaak een verschil gemaakt in precisie op basis van het aantal decimalen. Met bijvoorbeeld het getal 0,5 wordt eigenlijk een soort interval aangegeven: een waarde die tussen 0,45 en 0,54 ligt. Hoe meer decimalen, hoe preciezer deze getallen zijn. Het getal 0,50 geeft bijvoorbeeld een interval weer tussen 0,495 en 0,504; dit interval is met meer decimalen en daarmee is het getal preciezer. In RegelSpraak echter is dat niet het geval: een numerieke waarde in RegelSpraak is geen interval. 0,5 en 0,50 zijn twee volledig identieke waarden met precies dezelfde precisie. Het is goed om dit onderscheid te kennen en onderkennen.

Bij het weergeven van getallen zal RegelSpraak dan ook niet altijd het gespecificeerde aantal decimalen laten zien. De achterliggende nullen (zogenaamde achterloopnullen oftewel “trailing zero’s”) zullen niet getoond worden. Deze zijn in RegelSpraak ook niet nodig omdat alle numerieke waarden, zoals hierboven uitgelegd, toch dezelfde precisie hebben.

Stel dat we bijvoorbeeld een attribuut hebben met datatype *numeriek (geheel getal met 2 decimalen)*. Als we hier het resultaat van de berekening “1 gedeeld door 4” aan toekennen, dan zal dat gerepresenteerd worden met twee decimalen, oftewel als 0,25. Echter, als we hier het resultaat van “1 gedeeld door 2” aan toekennen, dan zal het gerepresenteerd worden met slechts één decimaal, oftewel als 0,5. Dit resultaat zou ook met 2 decimalen gerepresenteerd kunnen worden (als 0,50) maar de laatste nul is niet relevant bij deze waarde.

Het aantal decimalen van het resultaat van een berekening wordt hierbij bepaald door het aantal decimalen van de invoerwaarden van de berekening en door het soort berekening. Het

²⁰ Hierbij komen in basis alleen maar rationale getallen voor maar de benaming is in dit geval “getal”.

aantal decimalen van een optelling (de *plus* operator) zal bijvoorbeeld gelijk zijn aan het hoogste aantal decimalen van de waarden die worden opgeteld: 1,1 (met één decimaal) plus 1,11 (met twee decimalen) zal resulteren in 2,21 (met twee decimalen). In de volgende paragrafen zal dit bij elke soort berekening nader worden toegelicht.

Het is hier echter wel van belang of de invoerwaarden afkomstig zijn van een numerieke attribuut of parameter, of van een literal expressie:

- 1) Numerieke attributen of parameters hebben een expliciet gespecificeerd aantal decimalen. Het is bij het specificeren van een berekening in die situatie dan ook mogelijk om te garanderen dat het resultaat maximaal het gewenste aantal (gedefinieerde) decimalen bevat. Immers, dat aantal decimalen wordt bepaald door het soort berekening en het aantal decimalen van de invoerwaarden.
- 2) Een numerieke literal daarentegen heeft geen expliciet gespecificeerd aantal decimalen. Als een numerieke literal in een berekening wordt gebruikt, dan kan het aantal decimalen van het resultaat dus niet gegarandeerd worden.
Stel dat we een literal hebben met waarde 1000 en een numeriek attribuut met één decimaal. Het aantal decimalen van het resultaat zal dan afhangen van de waarde van het attribuut. Als die waarde van het numeriek attribuut 0,1 is, oftewel met één decimaal, dan is het resultaat van een deling door de literal met waarde 1000 gelijk aan 0,0001, oftewel met vier decimalen. Als de waarde van het numeriek attribuut echter 1000 is met nul decimalen, dan zal het resultaat 1 zijn, ook met nul decimalen. Dit kan tot onverwachte fouten leiden (de berekende 0,0001 zou immers een waarde zijn die niet past bij het eventueel gespecificeerde datatype met één decimaal), alert hierop zijn is dus belangrijk. Nogmaals, bij het gebruik van numerieke attributen of parameters komt dit niet voor aangezien het aantal decimalen daarvan altijd expliciet gespecificeerd is.

6.1.3 Afrondingen

In de vorige paragraaf 6.1.2 hebben we gezien dat resultaten van berekeningen soms afgerond moeten worden omdat ze anders teveel decimalen kunnen hebben om aan attributen toegekend te kunnen worden. Daarnaast kunnen er ook andere redenen zijn om het resultaat van een berekening af te ronden, bijvoorbeeld omdat een bepaalde regel dit voorschrijft (bedragen afronden op gehele getallen bijvoorbeeld). In deze paragraaf bespreken we hoe afronden in RegelSpraak in zijn werk gaat.

Afronden is het terugbrengen van het aantal decimalen naar een gewenst of noodzakelijk aantal. De laatste decimaal die moet blijven staan noemen we de **afrondpositie**. Het getal op de afrondpositie kan door afronding veranderen, afhankelijk van de manier waarop wordt afgerond. Er zijn vijf manieren beschikbaar in RegelSpraak om dit te doen:

- **naar beneden afronden**
Indien het getal positief is: het cijfer op de afrondpositie blijft hetzelfde.
Indien het getal negatief is: het cijfer op de afrondpositie wordt met 1 verhoogd indien een van de cijfers erna ongelijk is aan 0; als alle cijfers erna 0 zijn, dan blijft het cijfer op de afrondpositie hetzelfde.
- **naar boven afronden**
Indien het getal positief is: het cijfer op de afrondpositie wordt met 1 verhoogd indien een van de cijfers erna ongelijk is aan 0; als alle cijfers erna 0 zijn, dan blijft het cijfer op de afrondpositie hetzelfde.
Indien het getal negatief is: het cijfer op de afrondpositie blijft hetzelfde.

- **rekenkundig afronden**

Het cijfer na het cijfer op de afrondpositie geeft aan of hoe het cijfer op de afrondpositie wordt afgerond:

- indien het cijfer na het cijfer op de afrondpositie een 0, 1, 2, 3 of 4 is, dan blijft het cijfer op de afrondpositie hetzelfde;
- indien het cijfer na het cijfer op de afrondpositie een 5, 6, 7, 8 of 9 is, dan wordt het cijfer op de afrondpositie met 1 verhoogd.

- **richting nul afronden**

Het cijfer op de afrondpositie blijft in dit geval altijd hetzelfde, oftewel er wordt naar beneden afgerond (methode “naar beneden afronden”) indien het getal positief is en naar boven afgerond (methode “naar boven afronden”) indien het getal negatief is.

- **weg van nul afronden**

Het cijfer op de afrondpositie wordt met 1 verhoogd indien een van de cijfers erna ongelijk is aan 0; als alle cijfers erna 0 zijn, dan blijft het cijfer op de afrondpositie hetzelfde. Dit komt overeen met de methode “naar boven afronden” indien het getal positief is en de methode “naar beneden afronden” indien het getal negatief is.

In onderstaande tabel worden voorbeelden gegeven van de verschillende methodes van afronding. Hierbij is het cijfer na het cijfer op de afrondpositie onderstreept.

	Afgerond op aantal decimalen	Naar beneden	Naar boven	Rekenkundig	Richting nul	Weg van nul
12,33 <u>3</u> 6	2	12,33	12,34	12,33	12,33	12,34
12,33 <u>6</u> 0	2	12,33	12,34	12,34	12,33	12,34
- 12,33 <u>3</u> 6	2	- 12,34	- 12,33	- 12,33	- 12,33	- 12,34
- 12,33 <u>6</u> 6	2	- 12,34	- 12,33	- 12,34	-12,33	- 12,34
12,0 <u>3</u> 4	1	12,0	12,1	12,0	12,0	12,1
12, <u>3</u>	0	12	13	12	12	13
12,652 <u>3</u> 4	0	12	13	13	12	13
-12,0 <u>3</u> 4	1	-12,1	-12,0	-12,0	-12,0	-12,1
- 12,3 <u>5</u> 5	0	- 13	- 12	- 12	- 12	- 13
- 12,6 <u>5</u>	0	- 13	- 12	- 13	- 12	- 13
$\frac{1}{3}$	2	0,33	0,34	0,33	0,33	0,34
$-\frac{1}{3}$	2	- 0,34	- 0,33	- 0,33	- 0,33	- 0,34

Tabel 3: voorbeelden van de verschillende afrondingsmethodes.

Een belastingbedrag wordt bijvoorbeeld vaak afgerond op een geheel getal in het voordeel van de belastingplichtige. In het geval van de TOKA-casus betaalt de passagier een (positief) bedrag aan belasting. Afronden in het voordeel van de passagier betekent in dit geval dat het belastingbedrag *naar beneden* moet worden afgerond. Het onderstaande voorbeeld laat zien hoe dit er dan uit kan zien.

Regel Te betalen belasting van een passagier
geldig altijd

De te betalen belasting van een passagier moet gesteld worden op zijn belasting op basis van afstand naar beneden afgerond op 0 decimalen.

Syntax afronding specificatie:

<afrondingexpressie> ::= <getalexpressie> <afronding>


```
<afronding> ::= ("naar beneden" | "naar boven" | "rekenkundig" | "richting nul" | "weg van nul") "afgerond op" <geheel getal> "decimalen"
```

6.1.4 Begrenzen van waarden

Het resultaat van rekenkundige expressies met waarden met een *Numeriek* of *Percentage* datatype kan begrensd worden tot een bepaalde waarde.

Dit kan bijvoorbeeld gewenst zijn als het resultaat van een berekening niet negatief mag zijn. In dat geval kan het resultaat van de berekening op een minimum van 0 worden gesteld. Dit kan bijvoorbeeld het geval zijn bij het toepassen van een korting die maximaal zo groot mag zijn als het te betalen bedrag. Dit wordt uitgedrukt door aan de berekening toe te voegen **met een minimum van** gevolgd door een numerieke expressie (bijvoorbeeld een numerieke literal).

Maar ook kan het gewenst zijn om de uitkomst te begrenzen tot een maximum. Bijvoorbeeld in het geval van een boete die in de tijd oploopt, maar waarvoor een maximum bedrag geldt. Dit wordt uitgedrukt door **met een maximum van** gevolgd door een literal of numerieke expressie.

Het maximum en minimum kunnen ook in combinatie van toepassing zijn en in een regel worden opgenomen. Er kan maar één minimum en/of maximum worden gespecificeerd. Als het gewenst is om het minimum of maximum van meerdere waarden te specificeren, dan moet gebruik gemaakt worden van de aggregatie-expressie voor minimale/maximale waarde (zie paragraaf 5.8.3).

Voorbeeld van een begrenzing met een minimum:

Regel Te betalen belasting van een passagier
 geldig altijd
 De **te betalen belasting** van een **passagier** moet berekend worden als zijn **belasting op basis van afstand** min **korting bij gebruik niet-fossiele brandstof**, met een minimum van 0 € naar beneden afgerond op 0 decimalen.

Voorbeeld van begrenzing op zowel een minimum als een maximum:

Regel Te betalen belasting van een passagier
 geldig altijd
 De **te betalen belasting** van een **passagier** moet berekend worden als zijn **belasting op basis van afstand** min **korting bij gebruik niet-fossiele brandstof**, met een minimum van 0 € en een maximum van 1000 € naar beneden afgerond op 0 decimalen.

Syntax begrenzing specificatie:

```
<begrenzingexpressie> ::= <getalexpressie> "," <begrenzing>

<begrenzing> ::= (<begrenzingminimum> | <begrenzingmaximum> | <begrenzingminimum> "en"
<begrenzingmaximum>)

<begrenzingminimum> ::= "met een minimum van" <getalexpressie>

<begrenzingmaximum> ::= "met een maximum van" <getalexpressie>
```

6.2 Optellen: plus

Het optellen van twee numerieke waarden in RegelSpraak gebeurt met de operator **plus**. De operator *plus* werkt met twee numerieke expressies en levert als resultaat de optelling van de waarden uit deze expressies. Hierbij moet de eerste op te tellen expressie voor de operator staan en de tweede op te tellen expressie erna. Als we bijvoorbeeld 1 jaar willen optellen bij de *leeftijd* van een *passagier*, dan ziet dat er als volgt uit:

de *leeftijd* van een *passagier* plus 1 jr

Kanttekening hierbij is dat de gebruikte eenheden (zie paragraaf 3.7) van de expressies overeen moeten komen, of moeten via een eenheidssysteem in elkaar om te rekenen zijn. In het bovenstaande voorbeeld heeft de linkerexpressie de eenheid *jr* (jaar), aangezien het attribuut *leeftijd* van *Natuurlijk persoon* gespecificeerd is als *Numeriek (geheel getal) met eenheid jr*. De rechterexpressie moet ook deze eenheid hebben of een eenheid die met het eenheidssysteem *Tijd* omgerekend kan worden in *jr*. De expressie zou dus ook eenheden *mnd* (maand) of *kw* (kwartaal) kunnen hebben aangezien die naar elkaar omgerekend kunnen worden, maar niet de eenheden *euro* of *s*.

Het aantal decimalen van het resultaat van een optelling is gelijk aan het hoogste aantal decimalen van de gebruikte waarde. In de navolgende tabel is een aantal rekenkundige voorbeelden opgenomen.

Bij het bepalen van het aantal decimalen moet overigens rekening gehouden worden met de eventuele omrekenfactoren tussen eenheden. Deze zijn in onderstaande voorbeelden niet meegenomen.

Waarde links	Operator	Waarde rechts	Resultaat
1,01	plus	3,4	4,41
1,01	plus	3,43	4,44
1,01	plus	3,437	4,447
1,85	plus	1,05	2,9
1,85	plus	1,15	3

Tabel 4: het aantal decimalen van een optelling.

Het resultaat van de operator *plus* bij gebruik van lege waarden is te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *plus* een lege waarde behandeld wordt als zijnde het getal 0. Verder wordt alvast opgemerkt dat een plus-expressie anders omgaat met lege waarden dan de sommatie, zie hiervoor verder paragraaf 5.8.2 over deze specifieke aggregatiefunctie.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	plus <i>plus</i>	leeg <i>leeg</i>	= =	0 <i>0</i>
leeg <i>Voorbeeld: leeg</i>	plus <i>plus</i>	waarde <i>3</i>	= =	berekende waarde <i>3</i>
waarde <i>Voorbeeld: 6</i>	plus <i>plus</i>	leeg <i>leeg</i>	= =	berekende waarde <i>6</i>
waarde <i>Voorbeeld: 6</i>	plus <i>plus</i>	waarde <i>3</i>	= =	berekende waarde <i>9</i>

Tabel 5: lege waarden bij operator "plus".

De operator *plus* kan tot slot ook gebruikt worden om tijd bij een datum-tijd expressie op te tellen waardoor een nieuwe datum-tijd waarde ontstaat, dit staat in meer detail toegelicht in paragraaf 6.11.

Syntax berekening specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van operatoren die in navolgende paragrafen behandeld worden.

```
<berekening> ::= <getalexpressie> ("plus" | "min" | "verminderd met" | "maal" | "gedeeld door" | "gedeeld door (ABS)") <getalexpressie>
```

6.3 Aftrekken: min en verminderd met

Het aftrekken van twee numerieke waarden in RegelSpraak is mogelijk met de operatoren ***min*** en ***verminderd met***. Deze operatoren werken in basis op dezelfde manier, alleen gaan ze anders om met lege waarden. Beide operatoren werken met twee numerieke expressies: de eerste expressie moet voor de operator staan en de tweede erna. Het resultaat van de operator is de eerste expressie minus de tweede expressie. Als we bijvoorbeeld 1 jaar willen aftrekken van de *leeftijd* van een *passagier*, dan ziet dat er als volg uit:

de *leeftijd* van een *passagier* min 1 jr

Kanttekening hierbij is dat de gebruikte eenheden (zie paragraaf 3.7) van de expressies overeen moeten komen, of via een eenheidssysteem in elkaar om te rekenen moeten zijn. In het bovenstaande voorbeeld heeft de linkerexpressie de eenheid *jr* (jaar), aangezien het attribuut *leeftijd* van *Natuurlijk persoon* gespecificeerd is als *Numeriek (geheel getal) met eenheid jr*. De rechterexpressie moet ook deze eenheid hebben of een eenheid die met het eenheidssysteem *Tijd* omgerekend kan worden in *jr*. De expressie zou dus ook eenheden *mnd* (maand) of *kw* (kwartaal) kunnen hebben aangezien die naar elkaar omgerekend kunnen worden, maar niet de eenheden *euro* of *s*.

Het aantal decimalen van het resultaat van een aftrekking is gelijk aan het hoogste aantal decimalen van de gebruikte waarden. In de navolgende tabel is een aantal rekenkundige voorbeelden opgenomen.

Bij het bepalen van het aantal decimalen moet overigens rekening gehouden worden met omrekenfactoren tussen eenheden. Deze zijn in onderstaande voorbeelden niet meegenomen.

Waarde links	Operator	Waarde rechts	Resultaat
3,4	min/verminderd met	1,1	2,3
3,4	min/verminderd met	1,01	2,39
3,4	min/verminderd met	1,001	2,399
1,85	min/verminderd met	1,05	0,8
1,85	min/verminderd met	0,85	1

Tabel 6: het aantal decimalen van een aftrekking.

Het resultaat van de operator *min* bij gebruik van lege waarden is te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *min* een lege waarde behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	min <i>min</i>	leeg <i>leeg</i>	= <i>=</i>	0 <i>0</i>
leeg <i>Voorbeeld: leeg</i>	min <i>min</i>	waarde <i>4</i>	= <i>=</i>	berekende waarde <i>-4</i>
waarde <i>Voorbeeld: 5</i>	min <i>min</i>	leeg <i>leeg</i>	= <i>=</i>	berekende waarde <i>5</i>
waarde <i>Voorbeeld: 5</i>	min <i>min</i>	waarde <i>4</i>	= <i>=</i>	berekende waarde <i>1</i>

Tabel 7: lege waarden en aftrekken bij operator "min".

Het resultaat van de operator *verminderd met* bij gebruik van lege waarden is te zien in onderstaande tabel waarin wederom op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *verminderd met* het toepassen van een lege waarde afhankelijk is of de waarde links leeg is: in dat geval zal het resultaat altijd leeg zijn, onafhankelijk van de waarde rechts. Indien echter de waarde links niet leeg is, dan volgt er altijd een berekening waarbij een lege waarde behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	verminderd met <i>verminderd met</i>	leeg <i>leeg</i>	= <i>=</i>	leeg <i>leeg</i>
leeg <i>Voorbeeld: leeg</i>	verminderd met <i>verminderd met</i>	waarde <i>4</i>	= <i>=</i>	leeg <i>leeg</i>
waarde <i>Voorbeeld: 7</i>	verminderd met <i>verminderd met</i>	leeg <i>leeg</i>	= <i>=</i>	berekende waarde <i>7</i>
waarde <i>Voorbeeld: 7</i>	verminderd met <i>verminderd met</i>	waarde <i>4</i>	= <i>=</i>	berekende waarde <i>3</i>

Tabel 8: lege waarden en aftrekken bij operator "verminderd met".

De min operator kan tot slot ook gebruikt worden om tijd van een datum-tijd expressie af te halen waardoor een nieuwe datum-tijd waarde ontstaat, dit staat in meer detail toegelicht in paragraaf 6.11.

Syntax berekening specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van operatoren die in navolgende paragrafen behandeld worden.

```
<berekening> ::= <getalexpressie> ("plus" | "min" | "verminderd met" | "maal" | "gedeeld door" | "gedeeld door (ABS)") <getalexpressie>
```

6.4 Vermenigvuldigen: maal

Het vermenigvuldigen van twee numerieke waarden in RegelSpraak gebeurt met de operator *maal*. De operator *maal* werkt met twee numerieke expressies en levert als resultaat het product van de waarden uit deze expressies. De eerste expressie moet voor de operator staan en de tweede erna. Als we bijvoorbeeld de *te betalen belasting* van een *passagier* willen verdubbelen (maal 2), dan ziet dat er als volgt uit:

de *te betalen belasting* van een *passagier* maal 2

Bij vermenigvuldigen is het niet nodig dat de eenheden van de expressies overeenkomen of in elkaar zijn om te rekenen. Het resultaat zal altijd een samengestelde eenheid krijgen die bestaat uit het product van de eenheden van die expressies. Overigens zullen in de praktijk dit soort samengestelde eenheden niet zoveel voorkomen maar een voorbeeld is de eenheid voor hoeveelheid energie, de kiloWattuur (kWu²¹): als de linkerexpressie de eenheid kW (kiloWatt) heeft en de rechterexpressie de eenheid *u* (*uur*), dan is de eenheid van het resultaat van de vermenigvuldiging gelijk aan kW*u = kWu.

De eenheid van het resultaat van een vermenigvuldiging wordt verder altijd genormaliseerd waarbij gelijke eenheden boven en onder een eventuele deelstreep tegen elkaar worden weggestreept. Dus bijvoorbeeld 4 €/jaar maal 2 jaar = 8 € (de eenheid *jaar* is hierbij weggestreept, oftewel “verdwenen”).

Overigens geldt als kanttekening dat bij het vermenigvuldigen met de *maal* operator het niet mogelijk is gebruik te maken van *Percentage* expressies, oftewel de eenheid % kan niet gebruikt worden bij de operator *maal*.

Het aantal decimalen van het resultaat van een vermenigvuldiging is maximaal gelijk aan de som van het aantal decimalen van de gebruikte waarden. In de navolgende tabel is een aantal rekenkundige voorbeelden opgenomen. Bij het bepalen van het aantal decimalen moet overigens rekening gehouden worden met omrekenfactoren tussen eenheden. Deze zijn in onderstaande voorbeelden niet meegenomen.

Waarde links	Operator	Waarde rechts	Resultaat
1,1	maal	2,3	2,53
1,1	maal	2,31	2,541
1,1	maal	2,357	2,5927
2,85	maal	1,12	3,192
4,467	maal	1000	4467

Tabel 9: het aantal decimalen bij het vermenigvuldigen.

Tot slot is het resultaat van de operator *maal* bij gebruik van lege waarden te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *maal* een lege waarde behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	maal <i>maal</i>	leeg <i>leeg</i>	= =	0 <i>0</i>
leeg <i>Voorbeeld: leeg</i>	maal <i>maal</i>	waarde <i>4</i>	= =	0 <i>0</i>
waarde <i>Voorbeeld: 5</i>	maal <i>maal</i>	leeg <i>leeg</i>	= =	0 <i>0</i>
waarde <i>Voorbeeld: 5</i>	maal <i>maal</i>	waarde <i>4</i>	= =	berekende waarde <i>20</i>

Tabel 10: lege waarden bij operator “maal”.

²¹ De eenheid kiloWattuur wordt meestal afgekort tot kWh, waarbij de h van het Engelse woord hour afkomstig is. Om verwarring te voorkomen gebruiken we in het voorbeeld de letter u van het Nederlandse uur.

Syntax berekening specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van operatoren die in navolgende paragrafen behandeld worden.

```
<berekening> ::= <getalexpressie> ("plus" | "min" | "verminderd met" | "maal" | "gedeeld door" | "gedeeld door (ABS)") <getalexpressie>
```

6.5 Delen: gedeeld door en gedeeld door (ABS)

Het delen van twee numerieke waarden in RegelSpraak gebeurt met de operators **gedeeld door** en **gedeeld door (ABS)**²². Deze operatoren werken met twee numerieke expressies en leveren als resultaat het quotiënt van de waarden uit deze expressies. De eerste expressie moet voor de operator staan, de tweede erna. Als we bijvoorbeeld de *te betalen belasting* van een *passagier* willen halveren, dan ziet dat er als volgt uit:

De *te betalen belasting* van een *passagier* gedeeld door 2

Bij delen is het niet nodig dat de eenheden van de expressies overeenkomen of in elkaar zijn om te rekenen. Wel is het zo dat bij eenheden waarvoor dit niet geldt het resultaat een samengestelde eenheid zal krijgen die bestaat uit het quotiënt van de eenheden van de expressies. Als bijvoorbeeld de linkerexpressie de eenheid *euro* heeft en de rechterexpressie de eenheid *jr*, dan heeft het resultaat de eenheid *euro / jr*.

Net als bij de vermenigvuldiging wordt de eenheid van het resultaat van een deling altijd genormaliseerd waarbij gelijke eenheden boven en onder een eventuele deelstreep tegen elkaar worden weggestreept. Dus bijvoorbeeld 4 meter/seconde gedeeld door 2 meter = 2 seconde (de eenheid *meter* is hierbij “verdwenen”).

Bij de **gedeeld door (ABS)** operator wordt geen gebruik worden gemaakt van de breuknotatie zoals bij de **gedeeld door** operator. Hier is echter wel het aantal decimalen van het resultaat altijd maximaal 5 waarbij het resultaat richting 0 is/wordt afgerond. Hierdoor kan een resultaat van de **gedeeld door (ABS)** operator altijd in een bekende decimale notatie weergegeven worden.

Het aantal decimalen van het resultaat van een deling is gelijk aan het hoogste aantal decimalen van de gebruikte waarden. In de navolgende tabel is een aantal rekenkundige voorbeelden opgenomen.

Hierbij valt op dat als de deling niet op een mooi rond getal uitkomt, wat ook een getal kan zijn met een bepaald aantal decimalen, dan wordt het resultaat in breukvorm getoond.

Uitzondering hierop betreft de operator “gedeeld door (ABS)”, die toont het resultaat altijd met maximaal 5 decimalen zoals eerder besproken.

Bij het bepalen van het aantal decimalen moet overigens rekening gehouden worden met de eventuele omrekenfactoren tussen eenheden. Deze zijn in onderstaande voorbeelden niet meegenomen.

²² ABS staat voor AanslagBelastingenSysteem. Dit systeem bevatte een standaard afronding naar 0 op 5 decimalen van de resultaten uit een deling. Deze optie is gebaseerd op beleid uit het verleden.

Waarde link	Operator	Waarde rechts	Resultaat
2,3	gedeeld door	1,1	$2 \frac{1}{11}$
2,3	gedeeld door	1,12	$2 \frac{3}{56}$
2,3	gedeeld door	1,134	$2 \frac{16}{567}$
2,3	gedeeld door	1,15	2
2,3	gedeeld door (ABS)	1,1	2,0909
2,3	gedeeld door (ABS)	1,12	2,05357
2,3	gedeeld door (ABS)	1,134	2,02821
2,3	gedeeld door (ABS)	1,15	2
2	gedeeld door (ABS)	3	0,66667
2	gedeeld door (ABS)	-3	-0,66666

Tabel 11: aantal decimalen en gedeeld door/gedeeld door (ABS).

Tot slot is het resultaat van de operatoren *gedeeld door*, waarmee ook *gedeeld door (ABS)* impliciet wordt bedoeld, bij gebruik van lege waarden te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de beide operatoren een lege waarde behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	gedeeld door <i>gedeeld door</i>	leeg <i>leeg</i>	= =	0 ²³ <i>0</i>
leeg <i>Voorbeeld: leeg</i>	gedeeld door <i>gedeeld door</i>	waarde <i>4</i>	= =	0 <i>0</i>
waarde <i>Voorbeeld: 12</i>	gedeeld door <i>gedeeld door</i>	leeg <i>leeg</i>	= =	fout !!! <i>fout !!!</i>
waarde <i>Voorbeeld: 12</i>	gedeeld door <i>gedeeld door</i>	waarde <i>4</i>	= =	berekende waarde <i>3</i>

Tabel 12: lege waarden bij operator “gedeeld door” (en impliciet “gedeeld door (ABS)”).

Syntax berekening specificatie:

```
<berekening> ::= <getalexpressie> ("plus" | "min" | "verminderd met" | "maal" | "gedeeld door" | "gedeeld door (ABS)") <getalexpressie>
```

6.6 Worteltrekken: de wortel van

Worteltrekken in RegelSpraak gebeurt met de operator **de wortel van**. De operator werkt met één enkele numerieke expressie die achter de operator gezet moet worden. Aangezien de wortel van een getal niet per se een rationeel getal oplevert, is het verplicht om het resultaat van de wortheffing af te ronden (zie paragraaf 6.1.3). Als we bijvoorbeeld de wortel van de *leeftijd* van een *passagier* willen bepalen, dan ziet dat er als volgt uit:

de wortel van de *leeftijd* van een *passagier* naar beneden afgerond op 0 decimalen

²³ Wiskundig bekeken zal het delen door een lege waarde (waarde rechts) een fout moeten opleveren, in dit geval wordt het resultaat echter meteen op 0 gesteld zodra de waarde links gelijk is aan een lege waarde.

Syntax de wortel van specificatie:

```
<wortelfunctie> ::= "de wortel van" <getalexpressie> <afronding>
```

Twee kanttekeningen zijn van toepassing bij de operator *de wortel van*:

1. Aangezien RegelSpraak geen complexe getallen ondersteund zal er een foutmelding gegeven worden wanneer de numerieke expressie in een *de wortel van* operator een negatieve waarde bevat.
2. Als de expressie een lege waarde bevat, dan zal het resultaat ook een lege waarde zijn.

6.7 Machtsverheffen: tot de macht

Machtsverheffen in RegelSpraak gebeurt met de operator ***tot de macht***. De operator werkt met een numerieke expressie waarmee de vermenigvuldiging moet worden uitgevoerd die voor de operator gezet moet worden. Na de operator moet een numerieke expressie worden opgegeven die het aantal malen van vermenigvuldiging aangeeft. Aangezien het resultaat van machtsverheffen niet per se een rationeel getal is, is het verplicht om het resultaat van de machtsverheffenexpressie af te ronden (zie paragraaf 6.1.3). Als we bijvoorbeeld het kwadraat van de *woonregiofactor* van een *passagier* willen bepalen, dan ziet dat er als volgt uit:

het *kwadraat van de woonregiofactor* van een *passagier* moet berekend worden als zijn *woonregiofactor* tot de macht *2* rekenkundig afgerond op 0 decimalen

Syntax tot de macht specificatie:

```
<machtsverheffenfunctie> ::= <getalexpressie> "tot de macht" <getalexpressie> <afronding>
```

Twee kanttekeningen zijn van toepassing bij de operator *tot de macht*:

1. De numerieke expressies mogen geen expressie zijn met een eenheid.
2. Als één van de expressies een lege waarde bevat, dan zal het resultaat ook een lege waarde zijn.

6.8 Percentage bepalen: van

Om een percentage van een waarde te bepalen kan in RegelSpraak gebruik gemaakt worden van de operator ***van*** gecombineerd met een percentage expressie (dit betreft een expressie van het datatype percentage). De percentage-expressie dient voor de operator te staan, terwijl de expressie waarvan het percentage moet worden berekend achter de operator staat. Stel we hebben een parameter *btw tarief* waarin het te gebruiken percentage voor de btw staat. Verder hebben we een objecttype *product* met een attribuut *inkoopprijs*. Als we de btw zouden willen berekenen die voor een product afgedragen moet worden, dan ziet dat er als volgt uit:

het *BTW tarief* van een *inkoopprijs* van een *product*

Als het *btw tarief* 21 % is en de *inkoopprijs* 100 euro, dan zal het resultaat van bovenstaande expressie 21 euro zijn, en bij de *inkoopprijs van* 101 euro 21,21 euro zijn. Een percentageberekening is dus in basis een combinatie van een vermenigvuldiging met de waarde van de percentage-expressie en een deling door 100 waarbij het resultaat in basis verder niet afgerond wordt (tenzij dat dit expliciet is aangegeven, zie paragraaf 6.1.3).

Bij de *van* operator moeten overigens de eenheden van de resultaatexpressie en de expressie waarvan het percentage wordt genomen in elkaar om te rekenen zijn. De eenheid van het percentage is altijd ‘%’.

Tot slot is het resultaat van de *van* operator bij gebruik van lege waarden te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de *van* operator een lege waarde behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg%</i>	van <i>van</i>	leeg <i>leeg</i>	= <i>=</i>	0 <i>0</i>
leeg <i>Voorbeeld: leeg%</i>	van <i>van</i>	waarde <i>50</i>	= <i>=</i>	0 <i>0</i>
waarde <i>Voorbeeld: 10%</i>	van <i>van</i>	leeg <i>Leeg</i>	= <i>=</i>	0 <i>0</i>
waarde <i>Voorbeeld: 10%</i>	van <i>van</i>	waarde <i>50</i>	= <i>=</i>	berekende waarde <i>5</i>

Tabel 13: lege waarden en operator “van” (percentage bepalen).

Syntax van (percentage bepalen) specificatie:

```
<percentagefunctie> ::= <getalexpressie> ["%"] "van" <getalexpressie>
```

6.9 Absolute waarde van

Als het resultaat van een expressie mogelijk een negatieve numerieke waarde is, dan kan in een regel de absolute waarde van dat resultaat worden gebruikt. Die absolute waarde is een positieve waarde. RegelSpraak kent hiervoor de expressie “absolute waarde van”.

Bij de expressie “absolute waarde van” blijft het teken van het getal buiten beschouwing. De absolute waarde van -5 is gelijk aan 5, maar ook de absolute waarde van +5 is gelijk aan 5. De waarde van 0 blijft 0.

Een voorbeeld kan zijn dat van de te betalen belasting op basis van afstand een bedrag moet worden afgetrokken. Er is sprake van een correctie die altijd van het basisbedrag moet worden afgetrokken. Dit bedrag van de correctie is als negatief bedrag in de administratie vastgelegd. De regel kan er dan als volgt uitzien:

Regel vastgesteld inkomen 01

geldig altijd
De te betalen belasting op basis van afstand van een natuurlijk persoon moet berekend worden als zijn basisbedrag belasting op basis van afstand min de absolute waarde van (zijn correctie op belasting op basis van afstand).

Als in dit voorbeeld geen gebruik gemaakt wordt van “de absolute waarde van” dan wordt een negatief bedrag afgetrokken van het basisbedrag. Dat zou betekenen dat het erbij wordt opgeteld. Dat is niet de bedoeling. Door gebruik te maken van “de absolute waarde van” wordt de correctie wel afgetrokken van het basisbedrag.

De expressie kan alleen worden toegepast op expressies met een numeriek datatype.

Syntax absolute waarde van specificatie:

```
<absolutewaardefunctie> ::= "de absolute waarde van (" <getalexpressie> ")"
```

6.10 Tijdsduur van datum-tijd tot datum-tijd

In RegelSpraak kan de periode (tijdsduur) tussen twee datum-tijd expressies bepaald worden. Hierbij zal de volgende expressie voor tijdsduurberekening moeten worden gebruikt: “de tijdsduur van <datumexpressie 1> tot <datumexpressie 2> in hele <tijdseenheid>”. De datumexpressies 1 en 2 kunnen iedere expressie bevatten gelijk aan een datum-tijd. De expressie voor tijdsduurberekening berekent vervolgens de periode tussen de twee datumexpressies in een geheel getal met de tijdseenheid die in deze expressie gespecificeerd wordt waarbij het resultaat altijd naar beneden wordt afgerond. Van deze afronding is geen sprake bij de kleinst mogelijke tijdseenheid, millisecondes. In dat geval wordt de toevoeging “hele” weggelaten.

Voorbeelden zijn:

Regel Leeftijd passagier
 geldig altijd
 De **leeftijd** van een **passagier** moet berekend worden als de tijdsduur van zijn **geboortedatum** tot de **vluchtdatum** van zijn **reis** in hele jaren.

of

Regel Leeftijd passagier
 geldig altijd
 De **leeftijd** van een **passagier** moet berekend worden als de tijdsduur van zijn **geboortedatum** tot de **vluchtdatum** van zijn **reis** in millisecondes.

Daarnaast is het goed om te weten dat wanneer bij uitvoering van een dergelijke RegelSpraak-regel de waarde in datumexpressie 1 later is (in de tijd bekeken) dan de waarde in datumexpressie 2, als resultaat een negatieve tijdsduur wordt berekend.

Als het resultaat van de berekening positief moet zijn, ongeacht of de waarde van datumexpressie 1 later of eerder is dan de waarde van datumexpressie 2, dan kan gebruik gemaakt worden van de expressie “de absolute tijdsduur van <datumexpressie 1> tot <datumexpressie 2> in <tijdseenheid>”.

In onderstaand voorbeeld is het resultaat van de twee regels dan hetzelfde.

Regel Leeftijd passagier
 geldig altijd
 De **leeftijd** van een **passagier** moet berekend worden als de absolute tijdsduur van zijn **geboortedatum** tot de **vluchtdatum** van zijn **reis** in hele jaren.

Regel Leeftijd passagier
 geldig altijd
 De **leeftijd** van een **passagier** moet berekend worden als de absolute tijdsduur van de **vluchtdatum** van zijn **reis** tot zijn **geboortedatum** in hele jaren.

Tot slot is het resultaat van operator *tijdsduur van* of *absolute tijdsduur van* bij gebruik van lege waarden te zien in onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *tijdsduur van* een lege waarde behandeld wordt als zijnde “niet te gebruiken” waardoor het resultaat leeg blijft.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	tijdsduur van...tot... *) <i>tijdsduur van...tot... (in hele jaren)</i>	leeg <i>leeg</i>	= =	leeg <i>leeg</i>
leeg <i>Voorbeeld: leeg</i>	tijdsduur van...tot... *) <i>tijdsduur van...tot... (in hele jaren)</i>	waarde <i>1-1-2023</i>	= =	leeg <i>leeg</i>
waarde <i>Voorbeeld: 23-09-1970</i>	tijdsduur van...tot... *) <i>tijdsduur van...tot... (in hele jaren)</i>	waarde <i>leeg</i>	= =	leeg <i>leeg</i>
waarde <i>Voorbeeld: 23-09-1970</i>	tijdsduur van...tot... *) <i>tijdsduur van...tot... (in hele jaren)</i>	waarde <i>1-1-2023</i>	= =	berekende waarde <i>52</i>
waarde <i>Voorbeeld: 1-1-2023</i>	tijdsduur van...tot... *) <i>tijdsduur van...tot... (in hele jaren)</i>	waarde <i>23-9-1970</i>	= =	berekende waarde <i>-52</i>
waarde <i>Voorbeeld: 1-1-2023</i>	absolute tijdsduur van...tot... <i>tijdsduur van...tot... (in hele jaren)</i>	waarde <i>23-9-1970</i>	= =	berekende waarde <i>52</i>

Tabel 14: lege waarden en operator "tijdsduur van ... tot ...".

*) Geldt ook voor "absolute tijdsduur van ...tot...".

Syntax tijdsduur van datum tot datum specificatie:

```
<tijdsduurtussen> ::= ("de tijdsduur van " | "de absolute tijdsduur van ")
<datumexpressie> "tot" <datumexpressie> "in" ["hele"] <eenheidmeervoud>
```

6.11 Datum plus/min tijdseenheid

In RegelSpraak kunnen niet-lege waardes van een getalexpressie met een tijd-gerelateerde eenheid (dit is inclusief eventueel gedefinieerde dagsoorten) opgeteld of afgetrokken worden van een datumexpressie. Het resultaat is dan een nieuwe datumwaarde. Dit kan zowel later of eerder zijn dan de oorspronkelijke datum: optellen resulteert in een latere datumwaarde en aftrekken in een eerdere datumwaarde. Uitzondering betreft als de tijdseenheid gelijk is aan het getal 0, in dat geval zal de nieuwe datumwaarde gelijk zijn aan de oorspronkelijke datum.

Een voorbeeld hiervan is:

Regel verwachte datum-tijd van aankomst van een vlucht
geldig altijd

De *verwachte datum-tijd van aankomst* van een *vlucht* moet berekend worden als de *verwachte datum-tijd van vertrek* van de *vlucht* plus de *verwachte duur* van de *vlucht*.

Als gerekend wordt met een numerieke waarde met de eenheid van een specifieke dagsoort (bijvoorbeeld werkdagen) dan hoeft de datum waarbij een aantal werkdagen wordt opgeteld of afgetrokken zelf geen werkdag te zijn.

Als bijvoorbeeld 5 werkdagen bij datum X worden opgeteld dan is het resultaat de 5^e werkdag na datum X. Datum X telt dus niet mee, onafhankelijk van de vraag of het een werkdag is of niet. Het resultaat is altijd een datum die een werkdag is.

Tot slot is het resultaat van *datum plus/min tijdseenheid* bij gebruik van lege waarden te zien in

onderstaande tabel waarin ook op regelniveau een voorbeeld is opgenomen. Hierin is te zien dat bij de operator *datum plus/min tijdseenheid* een lege datumwaarde behandeld wordt als zijnde “niet te gebruiken” waardoor het resultaat leeg blijft, en een lege tijdseenheid behandeld wordt als zijnde het getal 0.

Waarde links	Operator	Waarde rechts	=	Resultaat
leeg <i>Voorbeeld: leeg</i>	datum plus/min tijdseenheid <i>datum plus tijdseenheid</i>	leeg <i>leeg jaar</i>	=	leeg <i>leeg</i>
leeg <i>Voorbeeld: leeg</i>	datum plus/min tijdseenheid <i>datum plus tijdseenheid</i>	waarde <i>12 jaar</i>	=	leeg <i>leeg</i>
waarde <i>Voorbeeld: 23-09-1970</i>	datum plus/min tijdseenheid <i>datum plus tijdseenheid</i>	leeg <i>leeg jaar</i>	=	berekende waarde <i>23-09-1970</i>
waarde <i>Voorbeeld: 23-09-1970</i>	datum plus/min tijdseenheid <i>datum plus tijdseenheid</i>	waarde <i>12 jaar</i>	=	berekende waarde <i>23-09-1982</i>

Tabel 15: lege waarden en datum plus/min tijdseenheid.

Syntax datum plus/min tijdseenheid specificatie:

```
<datumberekening> ::= <datumexpressie> ("plus" | "min") <getalexpressie>
<eenheidsafkorting>
```

6.12 Dag/maand/jaar uit

RegelSpraak bevat een patroon om de numerieke waarde van de dag, maand of jaar uit een datum te bepalen. Het resultaat is dan een positief geheel getal met de waarde van de dag, maand of jaar van de gespecificeerde datum. Een voorbeeld hiervan is ²⁴:

Regel Hoogseizoen

geldig altijd

Een vlucht is in het hoogseizoen

indien er aan ten minste één van de volgende voorwaarden wordt voldaan:

- de maand uit (de vluchtdatum van de vlucht) is gelijk aan 6
- de maand uit (de vluchtdatum van de vlucht) is gelijk aan 7
- de maand uit (de vluchtdatum van de vlucht) is gelijk aan 8.

Wanneer de datumexpressie in de berekening leeg is, zal het resultaat leeg zijn.

Syntax dag/maand/jaar uit specificatie:

```
<jaaruitfunctie> ::= "het jaar uit" <datumexpressie>
<maanduitfunctie> ::= "de maand uit" <datumexpressie>
<daguitfunctie> ::= "de dag uit" <datumexpressie>
```

²⁴ Merk op dat in het voorbeeld een voorwaardendeel wordt gebruikt. Het voorwaardendeel wordt behandeld in hoofdstuk 9.9.

6.13 Eerste paasdag van

RegelSpraak bevat een patroon voor het bepalen van de datum van de eerste paasdag van een jaar. Het jaartal wordt gespecificeerd door een geheel getal en het resultaat zal een datum in dagen zijn. Een voorbeeld hiervan is²⁵:

Regel Paaskorting
 geldig altijd
 Een vlucht is een reis met paaskorting
 indien de vluchtdatum van de vlucht gelijk is aan de eerste paasdag van (het jaar uit (de vluchtdatum van de vlucht)).

Wanneer het jaartal leeg is, wordt het resultaat van de eerste paasdag van de berekening ook op leeg gesteld.

Syntax de eerste paasdag van specificatie:

```
<eerstepaasdagvan> ::= "de eerste paasdag van (" <jaar> ")"
```

²⁵ Doordat dit een heel specifiek patroon wordt het voorbeeld erg bewerkelijk bij toepassing op de TOKA-casus.

7 Tijdsafhankelijke expressies

7.1 Aggregeren in de tijd: totaal van

De expressie “het totaal van” houdt binnen RegelSpraak een sommatie in van alle waarden van een attribuut in de tijd.

Voorwaarde voor het toepassen van deze expressie is dat het attribuut waarvan de waarden worden opgeteld tijdsafhankelijk is en een numeriek datatype heeft met een eenheid per tijdseenheid. Het resultaat is een niet-tijdsafhankelijke waarde.

Als bijvoorbeeld een passagier recht heeft op een maandelijks bedrag aan belastingvermindering, dan kan zijn totale belastingvermindering worden bepaald door alle maandelijks belastingverminderingen als volgt te sommeren:

Regel Totale belastingvermindering
 geldig altijd
 De *totale belastingvermindering* van een *passagier* moet gesteld worden op het totaal van zijn *belastingvermindering per maand*.

Het attribuut *belastingvermindering per maand* is hierbij een bedrag per maand dat voor elke maand wordt bepaald.

Het attribuut *totale belastingvermindering* is een bedrag en niet tijdsafhankelijk (heeft geen tijdlijn).

Als deze regel wordt toegepast op een situatie met een maandelijks belastingvermindering voor de passagier van:

- €10/maand van 1-1-2023 tot 1-1-2024 en
- €11/maand van 1-1-2024 tot 1-7-2024

dan is het resultaat voor zijn *totale belastingvermindering* €186 (12 maal €10 plus 6 maal €11).

Het is mogelijk om aan de expressie een tijdsafhankelijke conditie toe te voegen (zie paragraaf 8.4). In dat geval worden voor het bepalen van het resultaat van de expressie uitsluitend waarden gebruikt die van toepassing zijn in de periode zoals die in de tijdsafhankelijke conditie is gespecificeerd. N.B. De conditie heeft in dit geval alleen betrekking op de expressie en is geen onderdeel van het voorwaardendeel.

Voorbeeld van een regel met een tijdsafhankelijke conditie:

Regel Totale belastingvermindering
 geldig altijd
 De *totale belastingvermindering* van een *passagier* moet gesteld worden op het totaal van zijn *belastingvermindering per maand* gedurende de tijd dat *hij* een *recht op belastingvermindering* heeft.

Als de passagier recht op belastingvermindering heeft van 1-7-2023 tot 1-2-2024 dan is het resultaat voor zijn *totale belastingvermindering* €71 (6 maal €10 plus 1 maal €11).

Let op 1:

Als geen tijdsafhankelijke conditie wordt opgenomen of als die conditie geen beperkte periode betreft (“vanaf ...” of “... tot”) dan kan de expressie een onbeperkt aantal resultaten opleveren.

Let op 2:

De expressie moet tussen haakjes worden geplaatst of als variabele in de regel worden

opgenomen. De reden hiervoor is dat de regel anders ambigu wordt doordat niet duidelijk is of de tijdsafhankelijke conditie alleen de expressie betreft of een voorwaarde is.

Syntax waardepertijdseenheidaggregatie specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```
<waardepertijdseenheidaggregatie> ::= "het totaal van" <expressie>
[<conditiebijexpressie >]
```

7.2 Tellen van dagen: het aantal dagen in ... dat ...

Voor het bepalen van het aantal dagen dat in een periode aan een bepaalde voorwaarde is voldaan kan de expressie “het aantal dagen in ... dat ..” worden gebruikt.

In deze expressie moet na “het aantal dagen in” de tijdseenheid worden opgegeven waarover de telling moet plaatsvinden. Deze tijdseenheden zijn een maand of jaar uit het eenheidssysteem Tijd.

Het resultaat is een numerieke waarde voor een attribuut met als eenheid dagen per tijdseenheid en een tijdlijn met dezelfde tijdseenheid. Deze tijdseenheid komt overeen met de tijdseenheid die in de expressie is opgegeven (maand of jaar).

Achter de “dat” staat de voorwaarde. Alleen dagen waarop deze voorwaarde waar is worden meegeteld.

Als een passagier bijvoorbeeld wel of geen recht op belastingvermindering heeft op verschillende momenten in de tijd, dan kan per maand het aantal dagen dat een passagier recht op belastingvermindering heeft worden bepaald met de volgende regel:

Regel Aantal dagen recht op belastingvermindering
 geldig altijd
 Het aantal dagen recht op belastingvermindering van een passagier moet gesteld worden op het aantal dagen in de maand dat hij een recht op belastingvermindering heeft.

Als deze regel wordt toegepast op een situatie waarin de passagier recht op belastingvermindering heeft van:

- van 1-1-2024 tot 2-1-2024 en
- van 10-1-2024 tot 14-2-2024

dan is het resultaat voor het aantal dagen recht op belastingvermindering:

- 23 dagen/maand van 1-1-2024 tot 1-2-2024 en
- 13 dagen/maand van 1-2-2024 tot 1-3-2024

Het attribuut *aantal dagen recht op belastingvermindering* is hierbij gespecificeerd met een datatype Numeriek (geheel getal) met eenheid dagen/maand voor elke maand.

Let op: als de voorwaarde na de “dat” een zeer lange periode betreft dan kan de expressie een zeer groot aantal resultaten opleveren.

Syntax tellingaantaldagen specificatie:

```
<tellingaantaldagen> ::= "het aantal dagen in" ("de maand" | "het jaar") "dat"
<expressie>
```

7.3 Naar verhouding van tijdsduur**7.3.1 Standaard omrekening tussen jaren en maanden**

Op basis van het eenheidssysteem Tijd (zie paragraaf 3.7) worden met de in het eenheidssysteem gespecificeerde omrekenfactoren omrekeningen toegepast op waarden met verschillende tijdseenheden.

Zo kan bijvoorbeeld de waarde van een attribuut *belastingvermindering*, dat als datatype een numerieke waarde in € per maand heeft, worden afgeleid van een parameter STANDAARD BELASTINGVERMINDERING met een numerieke waarde in € per jaar. Dit kan worden uitgedrukt met de volgende regel:

Regel Belastingvermindering
geldig altijd

De *belastingvermindering* van een *passagier* moet gesteld worden op de STANDAARD BELASTINGVERMINDERING

Als in dit voorbeeld de STANDAARD BELASTINGVERMINDERING €120 €/jr is, dan is de afgeleide waarde van het attribuut *belastingvermindering* 10 €/mnd (120 gedeeld door de omrekenfactor 12 uit het eenheidssysteem).

Deze systematiek werkt voor zowel tijdsafhankelijke als niet-tijdsafhankelijk attributen en parameters.

Zoals in paragraaf 3.7 al toegelicht, is het binnen het eenheidssysteem Tijd niet mogelijk om vanuit jaren of maanden een omrekening toe te passen naar dagen. De reden daarvoor is dat het aantal dagen van een maand of jaar niet altijd gelijk is. Daardoor zijn hiervoor geen omrekenfactoren in het eenheidssysteem Tijd opgenomen.

7.3.2 Omrekening met gebroken jaren of maanden – tijdsevenredig deel

Met de expressie “het tijdsevenredig deel” kunnen regels worden opgesteld voor omrekeningen tussen jaar en maand waarbij sprake is van een gebroken jaar of maand. In die berekeningen wordt de omrekenfactor bepaald door per maand of jaar het aantal dagen waarop de expressie moet worden toegepast te delen door het totale aantal dagen van het jaar of de maand.

De (gebroken) periode waarover het tijdsevenredig deel moet worden berekend wordt uitgedrukt door aan de expressie een tijdsafhankelijke conditie toe te voegen (zie paragraaf 8.4).

Een voorbeeld van een regel waarin een tijdsevenredig deel wordt berekend op basis van een deel van een maand ziet er als volgt uit:

Regel Belastingvermindering
geldig altijd

De *belastingvermindering* van een *passagier* moet gesteld worden op het tijdsevenredig deel per maand van de STANDAARD BELASTINGVERMINDERING gedurende de tijd dat *hij* een recht op *belastingvermindering* heeft.

In de expressie wordt na “het tijdsevenredig deel per” gespecificeerd voor welke periode een tijdsevenredig deel moet worden berekend: een jaar of een maand. De knips in deze periode moeten passen op de tijdlijn van het attribuut waarvoor het resultaat wordt afgeleid (in het voorbeeld *de belastingvermindering*)

Daarna wordt gespecificeerd waarvan het tijdsevenredig deel moet worden berekend. In het voorbeeld is dat de parameter STANDAARD BELASTINGVERMINDERING.

Het deel van de regel na de parameter bevat de tijdsafhankelijke conditie met de specificatie van de periode waarvan de waarden gebruikt moeten worden voor de tijdsevenredige verdeling.

Let op:

De expressie moet tussen haakjes worden geplaatst of als variabele in de regel worden opgenomen. De reden hiervoor is dat de regel anders ambigu wordt doordat niet duidelijk is of de tijdsafhankelijke conditie alleen de expressie betreft of een voorwaarde is.

Voorbeeld 1 Afleiding per maand

Regel Belastingvermindering
 geldig altijd
 De *belastingvermindering* van een *passagier* moet gesteld worden op
 (het tijdsevenredig deel per maand van de STANDAARD BELASTINGVERMINDERING
 gedurende de tijd dat *hij* een *recht op belastingvermindering* heeft).

De regel geeft aan dat de tijdsevenredige delen per maand moeten worden afgeleid.

Het attribuut belastingvermindering heeft een numeriek datatype met eenheid €/mnd. Als tijdlijn van het attribuut is gespecificeerd dat de waarde elke maand kan wijzigen.

De parameter STANDAARD BELASTINGVERMINDERING heeft ook een numeriek datatype met eenheid €/mnd. De waarde van deze parameter kan jaarlijks wijzigen.

Het tijdsevenredig deel moet berekend worden over de waarde van de parameter STANDAARD BELASTINGVERMINDERING. Deze parameter heeft als waarde:

- 12 €/mnd van 1-1-2024 tot 1-1-2025
- 18 €/mnd vanaf 1-1-2025

De passagier heeft recht op belastingvermindering (bezittelijk kenmerk) van 1-1-2024 tot 8-2-2025.

Het resultaat van de regel worden dan de volgende afgeleide waarden voor het attribuut *belastingvermindering*:

- 12 €/mnd van 1-1-2024 tot 1-1-2025
- 18 €/mnd van 1-1-2025 tot 1-2-2025
- 4,5 €/mnd van 1-2-2025 tot 1-3-2025

In de regel is gespecificeerd dat waarden per maand moeten worden afgeleid. De periode van 1-1-2024 tot 1-2-2025 bestaat uit volledige maanden; er is geen omrekening van waarden nodig. De waarde van het attribuut is in die periode gelijk aan de waarde van de parameter. Voor de maand van 1-2-2025 tot 1-3-2025 geldt wel een omrekening. Deze maand bestaat uit 28 dagen en gedurende 7 dagen heeft de passagier recht op belastingvermindering. Het tijdsevenredig deel van de waarde van de parameter voor de hele maand (18 €/mnd) wordt 7/28 maal 18 €/mnd, dat is 4,5 €/mnd. Deze waarde geldt voor de hele maand van 1-2-2025 tot 1-3-2025 want waarden worden afgeleid per maand.

Voorbeeld 2 Afleiding per jaar

Als in afwijking van voorbeeld 1 de waarden niet per maand, maar per jaar moeten worden afgeleid dan ziet de regel er als volgt uit:

Regel Belastingvermindering
 geldig altijd
 De **belastingvermindering** van een **passagier** moet gesteld worden op
 (het tijdsevenredig deel per **jaar** van de **STANDAARD BELASTINGVERMINDERING**
 gedurende de tijd dat **hij** een **recht op belastingvermindering** heeft).

De overige specificaties en de gebruikte invoerwaarden zijn verdere gelijk aan voorbeeld 1:

De waarde van de parameter STANDAARD BELASTINGVERMINDERING:

- 12 €/mnd van 1-1-2024 tot 1-1-2025
- 18 €/mnd vanaf 1-1-2025

De passagier heeft recht op belastingvermindering (bezittelijk kenmerk) van 1-1-2024 tot 8-2-2025.

Het resultaat van de regel worden dan de volgende afgeleide waarden voor het attribuut *belastingvermindering*:

- 12 €/mnd van 1-1-2024 tot 1-1-2025
- $1 \frac{319}{365}$ €/mnd van 1-1-2025 tot 1-1-2026

In de regel is gespecificeerd dat waarden per jaar moeten worden afgeleid. De periode van 1-1-2024 tot 1-1-2025 bestaat uit een volledig jaar; er is geen omrekening van waarden nodig. De waarde van het attribuut is in die periode gelijk aan de waarde van de parameter. Voor het jaar van 1-1-2025 tot 1-1-2026 geldt wel een omrekening. Dit jaar bestaat uit 365 dagen en gedurende 38 dagen heeft de passagier recht op belastingvermindering. Het tijdsevenredig deel van de waarde van de parameter voor het hele jaar (18 €/mnd) wordt $38/365$ maal 18 €/mnd, dat is $1 \frac{319}{365}$ €/mnd. Deze waarde geldt voor het hele jaar van 1-1-2025 tot 1-1-2026.

Voorbeeld 3 Tijdsevenredige verdeling i.c.m. omrekening op basis van tijdseenheid

De totale berekening wordt complexer als naast de tijdsevenredige verdeling ook een omrekening op basis van de tijdseenheden van de datatypes van toepassing is.

Regel Belastingvermindering
 geldig altijd
 De **belastingvermindering** van een **passagier** moet gesteld worden op
 (het tijdsevenredig deel per **maand** van de **STANDAARD BELASTINGVERMINDERING**
 gedurende de tijd dat **hij** een **recht op belastingvermindering** heeft).

Dit is dezelfde regel als uit voorbeeld 1.

Maar in afwijking op voorbeeld 1 heeft de parameter STANDAARD BELASTINGVERMINDERING heeft een numeriek datatype met eenheid €/jr. De waarde van deze parameter kan jaarlijks wijzigen zoals ook in voorbeeld 1 het geval is.

De invoerwaarden zijn voor de waarde van de parameter STANDAARD BELASTINGVERMINDERING:

- 120 €/jr van 1-1-2024 tot 1-1-2025

- 180 €/jr vanaf 1-1-2025

De passagier heeft recht op belastingvermindering (bezittelijk kenmerk) van 1-1-2024 tot 8-2-2025.

Het resultaat van de regel worden dan de volgende afgeleide waarden voor het attribuut *belastingvermindering*:

- 10 €/mnd van 1-1-2024 tot 1-1-2025
- 15 €/mnd van 1-1-2025 tot 1-2-2025
- 3,75 €/mnd van 1-2-2025 tot 1-3-2025

Op basis van het eenheidssysteem vindt een omrekening plaats van de eenheid van de parameter (€/jr) naar de eenheid van het attribuut waarvoor de waarde wordt afgeleid (€/mnd). Dat betekent dat de waarde van de parameter wordt gedeeld door 12 om de eenheid per jaar om te zetten naar een eenheid per maand:

- 10 €/mnd van 1-1-2024 tot 1-1-2025
- 15 €/mnd vanaf 1-1-2025

In de regel is gespecificeerd dat waarden per maand moeten worden afgeleid. De periode van 1-1-2024 tot 1-2-2025 bestaat uit volledige maanden; er is geen verdere omrekening van waarden nodig. De waarde van het attribuut is in die periode gelijk aan de waarde van de parameter. Voor de maand van 1-2-2025 tot 1-3-2025 geldt wel een omrekening. Er wordt gerekend met het aantal dagen volgens de kalender. Deze maand bestaat uit 28 dagen en gedurende 7 dagen heeft de passagier recht op belastingvermindering. Het tijdsevenredig deel van de waarde van de parameter voor de hele maand (15 €/mnd) wordt $\frac{7}{28}$ maal 15 €/mnd, dat is 3,75 €/mnd. Deze waarde geldt voor de hele maand van 1-2-2025 tot 1-3-2025.

Syntax tijdsevenredigdeel specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```
<tijdsevenredigdeel> ::= "het tijdsevenredig deel per" ("maand" | "jaar") "van"
<expressie> <conditiebijexpressie>
```

8 Condities en predicaten

In paragraaf 4.3 is het voorwaardendeel van een regel besproken, dat de voorwaarden beschrijft waaraan voldaan moet zijn om de actie in het resultaatdeel van de regel uit te voeren. In paragraaf 9.5 zullen consistentieregels besproken worden waarbij sprake is van criteria waaraan voldaan moet worden. Zowel criteria als voorwaarden worden in RegelSpraak **condities** genoemd.

Condities worden gespecificeerd met een predicaat. Een **predicaat** is een RegelSpraak uitdrukking die iets controleert en “waar” oplevert als de controle slaagt en “onwaar” als de controle niet slaagt. De meeste predicaten worden achter een onderwerpexpressie gezet en controleren of een rol, attribuut of expressie ergens aan voldoen. Uitzonderingen zijn de predicaten *is gevuurd* en *is inconsistent* die de uitvoering van een regel controleren in plaats van een rol, attribuut of expressie; bij deze predicaten is dan ook geen onderwerpexpressie nodig maar zal een regelversie opgegeven moeten worden.

Een predicaat heeft meerdere schrijfwijzen. Dit is afhankelijk van waar en hoe het predicaat gebruikt wordt. Predicaten hebben zowel een vragende als een stellende vorm, die beide een enkelvoud- en een meervoudsvorm hebben. Welke vorm gebruikt wordt hangt van een aantal zaken af. In dit hoofdstuk zullen de predicaten en hun schrijfwijzen verder beschreven worden.

8.1 Predicaten

Er bestaat een aantal verschillende predicaten in RegelSpraak:

1. Twee van deze predicaten controleren of een bepaalde regel een resultaat heeft opgeleverd (*is gevuurd* en *is inconsistent*) en moeten dan ook gecombineerd worden met de desbetreffende regelversie (zie paragraaf 4.2).
2. Twee andere predicaten controleren het al dan niet aanwezig zijn van een kenmerk (*is/heeft/is een* en *is niet/heeft geen/is geen*), en moeten gecombineerd worden met de naam van het objecttype en het kenmerk.
3. Nog eens twee andere predicaten controleren verder of een instantie ook een bepaalde rol invult (*is/heeft* en *is geen/heeft geen*) en moeten gecombineerd worden met de namen van de rol en van het objecttype.
4. De overige predicaten controleren een waarde. Deze waarde is afkomstig van een onderwerpexpressie die voor het predicaat gezet moet worden. Hierbij is het datatype van de onderwerpexpressie van belang aangezien niet elk predicaat bij elk datatype gebruikt kan worden. Voor sommige van deze predicaten, de vergelijkingspredicaten, is een tweede onderwerp vereist dat achter het predicaat moet staan.

Zoals al eerder gezegd hebben predicaten een vragende en een stellende vorm, die beide een enkelvoud- en meervoudsvorm kennen. De onderstaande tabel toont de predicaten met deze vormen, en met de datatypes die erbij moeten worden gebruikt. Deze predicaten zullen in de volgende subparagrafen verder worden besproken.

Soorten predicaten							
			Datatype				
	Vragende vorm	Stellende vorm	Numeriek/ Percentage	Datum- tijd	Tekst	Boolean	Enumeratie waarde
Vergelijking							
	gelijk is aan gelijk zijn aan	is gelijk aan zijn gelijk aan	X	X	X	X	X
	ongelijk is aan ongelijk zijn aan	is ongelijk aan zijn ongelijk aan	X	X	X	X	X
	groter is dan groter zijn dan	is groter dan zijn groter dan	X				
	groter of gelijk is aan groter of gelijk zijn aan	is groter of gelijk aan zijn groter of gelijk aan	X				
	kleiner is dan kleiner zijn dan	is kleiner dan zijn kleiner dan	X				
	kleiner of gelijk is aan kleiner of gelijk zijn aan	is kleiner of gelijk aan zijn kleiner of gelijk aan	X				
	later is dan later zijn dan	is later dan zijn later dan		X			
	later of gelijk is aan later of gelijk zijn aan	is later of gelijk aan zijn later of gelijk aan		X			
	eerder is dan eerder zijn dan	is eerder dan zijn eerder dan		X			
	eerder of gelijk is aan eerder of gelijk zijn aan	is eerder of gelijk aan zijn eerder of gelijk aan		X			
Lege waarden							
	leeg is leeg zijn	is leeg zijn leeg	X	X	X	X	X
	gevuld is gevuld zijn	is gevuld zijn gevuld	X	X	X	X	X
Elfproef							
	aan de elfproef voldoen	voldoet aan de elfproef	X		X		
	niet aan de elfproef voldoen	voldoet niet aan de elfproef	X		X		
Getalcontrole							
	numeriek met exact <geheelgetal> cijfers is	is numeriek met exact <geheelgetal> cijfers			X		
	niet numeriek met exact <geheelgetal> cijfers is	is niet numeriek met exact <geheelgetal> cijfers			X		
Dagsoortcontrole							
	een <dagsoort> is	is een <dagsoort>		X			
	geen <dagsoort> is	is geen <dagsoort>		X			
Uniciteit²⁶							
	uniek zijn		X	X	X	X	X
Rolcheck²⁷							
	<rolnaam> is/heeft <rolnaam> zijn/hebben	Is/heeft <rolnaam> Zijn/hebben <rolnaam>					
	geen <rolnaam> is/heeft geen <rolnaam>	Is/heeft geen <rolnaam> zijn/hebben geen					

²⁶ Uniciteit predicaten kunnen alleen gebruikt worden in consistentieregels, en slechts in een elementair criterium. Er is dan ook slechts 1 vorm.

²⁷ Rolcheck predicaten controleren instanties van objecttypen. Het datatype is hier niet van belang, aangezien het onderwerp een verwijzing naar een objecttype is.

	zijn/hebben	<rolnaam>	
Kenmerkcheck²⁸			
	is/heeft/is een <kenmerknaam>	is/heeft/ is een <kenmerknaam>	
	zijn/hebben/zijn een <kenmerknaam>	zijn/hebben/zijn een <kenmerknaam>	
	geen <kenmerknaam> is /heeft / is geen <kenmerknaam>	is niet/heeft niet/is niet een <kenmerknaam>	
	zijn /hebben /zijn	zijn geen/hebben geen/zijn geen <kenmerknaam>	
Resultaat Regel²⁹			
	gevuurd is	is gevraagd	
	inconsistent is	is inconsistent	

Tabel 16: de predicaten in RegelSpraak en de datatypes die erbij gebruikt moeten worden.

N.B. Voor de voorbeelden in de navolgende subparagrafen zal de enkelvoudige, stellende vorm worden gebruikt, de tweede kolom in bovenstaande tabel. De voorbeelden zijn bedoeld om het te bespreken predicaat te illustreren, en zullen derhalve dan ook geen complete regels bevatten maar slechts condities waar het predicaat in wordt gebruikt.

8.1.1 Vergelijkingen

Vergelijkingspredicaten vergelijken de waarde die uit de onderwerpexpressie komt met de waarde die uit een tweede expressie komt. Die tweede expressie moet hierbij hetzelfde datatype hebben als de onderwerpexpressie: zo kan een datum bijvoorbeeld niet vergeleken worden met een getal. Indien de waarden gebruik maken van verschillende eenheden uit een eenheidssysteem (zie paragraaf 3.7), dan zal de waarde uit de tweede expressie omgerekend worden naar de eenheid van de eerste expressie voordat de vergelijking plaatsvindt.

Twee vergelijkingspredicaten kunnen controleren of waardes wel of niet precies aan elkaar gelijk zijn. Deze predicaten zijn bij elk datatype te gebruiken. Als we bijvoorbeeld willen weten of een *Vlucht* een rondvlucht is, dan zouden we de *luchthaven van vertrek* en de *luchthaven van bestemming* kunnen vergelijken:

- de *luchthaven van vertrek* van de *vlucht* is gelijk aan de *luchthaven van bestemming* van de *vlucht*

Wanneer een van de expressies een lege waarde bevat geeft het *is gelijk* predicaat *onwaar* als resultaat en geeft het *is niet gelijk* predicaat *waar* als resultaat. Wanneer beide expressies een lege waarde bevatten geeft het *is niet gelijk* predicaat *onwaar* als resultaat en geeft het *is gelijk* predicaat ook *onwaar* als resultaat in het geval van expressies met het datatype Numeriek of Percentage en geeft dit predicaat een *foutmelding* in het geval van expressies met andere soorten datatypes.

Verder zijn er vier vergelijkingspredicaten die alleen gebruikt kunnen worden bij *numerieke* en *percentage* datatypes. Deze predicaten controleren of de ene waarde groter of kleiner is dan de andere. Als we bijvoorbeeld willen weten of een reis onder de eerste schijf valt, dan zouden we

²⁸ Kenmerkcheck predicaten controleren instanties van objecttypen. Hun onderwerpexpressie is de naam van het desbetreffende objecttype. Het datatype is hier niet van belang. Aangezien er drie soorten kenmerken zijn, zijn er drie verschillende schrijfwijzen per soort conditie.

²⁹ Regelcontrole predicaten controleren regels. Hun invoer is de regelversie, d.w.z. de naam en de geldigheid van de desbetreffende regel. Het datatype is hier niet van belang.

kunnen controleren of de *afstand tot bestemming* kleiner is dan de parameter *bovengrens afstand eerste schijf*:

- de *afstand tot bestemming* van de *vlucht* is kleiner dan de *bovengrens afstand eerste schijf*

Wanneer een van beide of beide expressies een lege waarde bevat zal het resultaat van het predicaat *onwaar* zijn.

Als laatste zijn er vier vergelijkingspredicaten die alleen gebruikt kunnen worden bij *datum/tijd* datatypes. Deze controleren of de ene datum voor of na de andere datum ligt. Als we bijvoorbeeld willen controleren of de *uiterste boekingsdatum* van een *Vlucht* wel voor de *datum van vertrek* van de *Vlucht* zelf ligt, dan ziet de vergelijking er als volgt uit:

- de *uiterste boekingsdatum* van de *vlucht* is eerder dan de *datum van vertrek* van de *vlucht*

Wanneer een beide expressies onwaar is zal het predicaat het resultaat *onwaar* geven en wanneer beide waarden leeg zijn geeft het predicaat een *foutmelding* als resultaat.

8.1.2 Lege waarden

Met de predicaten *leeg zijn/leeg is/is leeg* en *gevuld zijn/gevuld is/is gevuld* kan gecontroleerd worden of een attribuut al dan niet leeg is, oftewel of het al dan niet een waarde heeft. Om bijvoorbeeld te controleren of de *leeftijd* van een *Natuurlijk persoon* een waarde heeft, gebruiken we de volgende conditie:

- zijn *leeftijd* is gevuld

8.1.3 De elfproef

Een speciaal soort predicaat is de *aan de elfproef voldoen*. De hier bedoelde elfproef (er zijn verschillende varianten voor verschillende doeleinden) is een test voor het beoordelen van de geldigheid van een identificatienummer zoals met name het Burgerservicenummer (BSN-nummer). Hierbij wordt ieder cijfer “gewogen” en bij elkaar opgeteld. Gewogen betekent dat een cijfer vermenigvuldigd wordt met een afgesproken waarde, afhankelijk van zijn positie in de cijferreeks. Het toekennen van een weging begint bij het laatste cijfer in de reeks: het laatste cijfer in de reeks (het controlecijfer) krijgt een gewicht van -1 , het een-na-laatste cijfer een gewicht van 2 , het twee-na-laatste cijfer een gewicht van 3 , etc. Als de gebruikte cijferreeks een geldig BSN-nummer is, moet de som hiervan bij delen door 11 (vandaar de naam “elfproef”) een positief geheel getal opleveren en het getal mag niet uitsluitend uit nullen bestaan.

De genoemde weging van cijfers bij het toepassen van de elfproef impliceert dat slechts enkele datatypes mogelijk zijn: het datatype Numeriek (geheel getal) en Tekst zijn toegestaan, met als kanttekening dat bij het datatype Tekst alleen cijfers in de waarde mogen voorkomen (dit kan bijvoorbeeld het geval zijn bij het gebruik van eventuele voor- of achterloopnullen).

Om de werking van de elfproef te verduidelijken, is hierna een uitgewerkt voorbeeld gegeven aan de hand van fictief BSN-nummer 192837465. Eerst wordt zoals hierboven aangegeven voor ieder cijfer een gewicht toegekend, waarna het cijfer met zijn gewicht wordt vermenigvuldigd. De som van deze producten wordt berekend en vervolgens gedeeld door 11 . De som is 205 en delen door 11 geeft 18 rest 7 . Dit betekent dat het fictief BSN-nummer 192837465 geen geldig burgerservicenummer is.

BSN	1	9	2	8	3	7	4	6	5
Gewicht	9	8	7	6	5	4	3	2	-1
Product	9	72	14	48	15	28	12	12	-5
Som	9	81	95	143	158	186	198	210	205

In RegelSpraak kunnen we de volgende voorwaarde schrijven om de elfproef uit te voeren:

- zijn **burgerservicenummer** voldoet aan de elfproef

8.1.4 Getalcontrole

Het predicaat *numeriek zijn met exact <geheel getal> cijfers* kan alleen gebruikt worden bij een *Tekst* datatype. Het predicaat controleert of de tekst ook gelezen kan worden als een getal, oftewel als een numerieke waarde met het opgegeven aantal cijfers.

We zouden bijvoorbeeld naast de elfproef een andere controle op het burgerservicenummer kunnen doen. Een burgerservicenummer bestaat uit 9 cijfers, en met dit predicaat kunnen we controleren of dat ook klopt:

- zijn **burgerservicenummer** is numeriek met exact 9 cijfers

8.1.5 Dagsoortcontrole

Het predicaat *is een <dagsoort>* kan alleen gebruikt worden bij een *datum/tijd* datatype. Zoals uitgelegd in paragraaf 3.12 is het in RegelSpraak mogelijk om dagsoorten te definiëren. Met het predicaat *is een <dagsoort>* kan vervolgens gecontroleerd worden of een bepaalde datum (dag) ook tot een bepaalde dagsoort behoort.

Als we bijvoorbeeld een uitzondering willen maken voor vluchten die tijdens kerstmis vertrekken, dan kunnen we de volgende conditie gebruiken:

- de **datum van vertrek** van een **vlucht** is een kerstdag

8.1.6 Unicité

Een speciaal soort predicaat is *uniek zijn*. Met dit predicaat kan voor één of een combinatie van attributen gecontroleerd worden of deze over de instanties heen dezelfde waarde hebben. Dit noemen we **uniciteit**. Is de waarde van een attribuut of combinatie van attributen uniek over instanties heen dan heeft het predicaat *is uniek* de waarde 'waar'. Dit predicaat kan alleen gebruikt worden in consistentieregels, en alleen in een enkel criterium (dus niet in een samengesteld criterium). De attributen waarvan de waardes worden vergeleken moeten hierbij gespecificeerd worden.

Bijvoorbeeld, bij instanties van het objecttype *Natuurlijk persoon* zal uniciteit moeten gelden met betrekking tot het attribuut *Burgerservicenummer*. Immers, elke persoon heeft een uniek Burgerservicenummer.

In RegelSpraak wordt ervan uitgegaan dat de verschillende instanties van een objecttype ook verschillende objecten in de realiteit weergeven. Uniciteitschecks zijn alleen nodig als dat niet het geval is. De objecten zijn dan eerdere representaties van (of gegevens over) deze werkelijke objecten in verschillende contexten. Om dan te kunnen achterhalen dat twee representaties over hetzelfde werkelijke object gaan is er een identificerend attribuut nodig waarvan de

uniciteit per werkelijk object gewaarborgd wordt.

Als we de uniciteit willen controleren, dan moeten we gebruik maken van een speciale onderwerpexpressie. In plaats van een lidwoord moeten we het woord *alle* voor het desbetreffende objecttype zetten.

- de *burgerservicenummers* van alle *Natuurlijke personen* moeten uniek zijn

Uniciteit kan verder ook gecontroleerd worden voor een combinatie van attributen. Er wordt dan gecontroleerd of de combinatie van de attribuutwaarden uniek is. Een consistentieregel met twee of meer attributen schrijven we door een concatenatie (zie ook paragraaf 5.7) van deze twee attributen te specificeren. Twee vluchten bijvoorbeeld kunnen niet tegelijkertijd van dezelfde luchthaven vertrekken, zoals in het onderstaande voorbeeld:

- de concatenatie van de *luchthaven van vertrek* en de *vertrektijd van de vlucht* van alle *Vluchten* is uniek

Uniciteit kan ook gecontroleerd worden met betrekking tot de attributen van twee verschillende objecttypen. Dit is wat lastig uit te leggen met de TOKA-casus dus we zullen hiervoor een abstracter voorbeeld gebruiken. Stel we hebben de volgende twee objecttypen voor twee verschillende vormen:

```
Objecttype de Cirkel (mv: Cirkels)
  de naam                Tekst;
  de kleur               Tekst;
  de radius              Numeriek (getal met 2 decimalen);

Objecttype het Vierkant (mv: Vierkanten)
  de naam                Tekst;
  de kleur               Tekst;
  de afmeting            Numeriek (getal met 2 decimalen);
```

We willen zowel een *Cirkel* als *Vierkant* kunnen aanwijzen met de naam. We kunnen dan controleren of er geen vormen zijn met dezelfde naam, oftewel

1. of er geen twee *Cirkels* zijn met dezelfde naam,
2. of er geen twee *Vierkanten* zijn met dezelfde naam, en
3. of er geen *Cirkel* met dezelfde naam is als een *Vierkant*.

Dit doen we door een **vereniging** te specificeren van de twee attributen. Let wel: een vereniging is de combinatie van twee attributen met hetzelfde datatype. Het attribuut *naam* heeft het datatype *Tekst* voor zowel *Cirkel* als *Vierkant*; we kunnen deze combineren door gebruik te maken van de operator *verenigd met*. Het resultaat hiervan is een verzameling die bestaat uit de waarden van *naam* voor alle instanties van beide objecttypen:

- de *namen* van alle *Cirkels* verenigd met de *namen* van alle *Vierkanten* moeten uniek zijn

Tenslotte kan de uniciteit gecontroleerd worden voor een combinatie van twee attributen van twee objecttypen. We kunnen dan bijvoorbeeld controleren of er geen figuren zijn met dezelfde combinatie van naam en kleur. We specificeren dan twee concatenaties van deze attributen voor beide objecttypen die we gebruiken voor de vereniging waarvoor vervolgens de uniciteit bepaald moet worden:

- de concatenatie van de *namen* en de *kleuren* van alle *Cirkels* verenigd met de concatenatie van de *namen* en de *kleuren* van alle *Vierkanten* moeten uniek zijn

Belangrijk hierbij is dat de volgorde van de gebruikte attributen voor beide concatenaties hetzelfde moet zijn. In het bovenstaande voorbeeld is dat het geval: het eerste attribuut is *naam* en het tweede attribuut is *kleur*, voor zowel het objecttype *Cirkel* als het objecttype *Vierkant*. Als dit niet het geval zou zijn, dan zou RegelSprak een naam/kleur combinatie vergelijken met een kleur/naam combinatie, en geen uniciteit vaststellen bij gelijke *naam* en

kleur. Immers, “blauw/eerste vorm” is niet gelijk aan “eerste vorm/blauw”.

Syntax consistentieregel voor uniciteit specificatie:

```
<uniciteitscontrole> ::= (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>)
<vereniging>* "moeten uniek zijn."

<vereniging> ::= "verenigd met" (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>)

<alleattribuutvanonderwerp> ::= "de" <meervoudsvorm> "van alle" (objecttypenaam |
<rolnaam> "van" <onderwerpketen>)) ["van" <onderwerpketen>]

<uniciteitconcatenatie> ::= "de concatenatie van" <meervoudsvorm> ("," <meervoudsvorm>)*
"en" <meervoudsvorm> "van alle" (objecttypenaam | <rolnaam> "van" <onderwerpketen>))
["van" <onderwerpketen>]
```

8.1.7 Rolcheck

Met een **rolcheck** kan gecontroleerd worden of een instantie al dan niet een bepaalde rol (zie paragraaf 3.11) invult. Hiervoor worden de predicaten *is/heeft een of is/heeft geen* gebruikt waar de te controleren rol achter wordt gezet.

Als we bijvoorbeeld willen controleren of een instantie van *Natuurlijk persoon* ook de rol *passagier* invult bij de instanties van *Vlucht*, dan gebruiken we de volgende conditie:

- hij is een *passagier*

De controle of er *natuurlijke personen* zijn die *passagier* zijn kan worden uitgedrukt met “heeft” in combinatie met het andere objecttype in het feittype. Bijvoorbeeld:

- de *Vlucht* heeft een *passagier*

8.1.8 Kenmerkcheck

Met een **kenmerkcheck** kan gecontroleerd worden of een instantie al dan niet een bepaald kenmerk (zie paragraaf 3.5) heeft. Aangezien een kenmerk bijvoeglijk, bezittelijk of geen van beide kan zijn hebben de predicaten verschillende schrijfwijzen.

1. Bij bijvoeglijke kenmerken wordt *is <kenmerknaam>* of *is niet <kenmerknaam>* gebruikt.
2. Bij bezittelijke kenmerken wordt *heeft <kenmerknaam>* of *heeft geen <kenmerknaam>* gebruikt.
3. Bij een kenmerk dat niet bijvoeglijk of bezittelijk is, wordt *is een <kenmerknaam>* of *is geen <kenmerknaam>* gebruikt.

Voor een volledig beeld van de verschillende schrijfwijzen kan Tabel 16 geraadpleegd worden.

Het predicaat dient voorafgegaan te worden door een onderwerpexpressie die verwijst naar een objecttype (eventueel via een rol). De onderwerpexpressie kan niet verwijzen naar een datatype (via een attribuut) aangezien datatypes geen kenmerken hebben. Als we bijvoorbeeld willen controleren of een *Natuurlijk persoon* minderjarig is, dan gebruiken we de volgende conditie:

- hij is *minderjarig*

8.1.9 Regelpredicaat

Regelpredicaten kunnen gebruikt worden om de uitvoering van een regel te controleren. Er zijn twee soorten controles, te weten *controle op consistentie* en *controle op vuren*.

1. Consistentieregels controleren of attributen, rollen of instanties aan bepaalde criteria

voldoen, en geven als resultaat **inconsistent** als dat niet het geval is. In paragraaf 9.5 worden consistentieregels verder uitgewerkt. Het regelpredicaat *is inconsistent* kan hierbij gebruikt worden om het resultaat van een consistentieregel te gebruiken als conditie. Er is geen onderwerpexpressie nodig bij dit predicaat; in plaats daarvan moet het voorafgegaan worden door de desbetreffende regelversie.

Stel dat we een consistentieregel hebben gespecificeerd, *Controleer of vlucht geen rondvlucht is*³⁰, die controleert of de *luchthaven van vertrek* ongelijk is aan de *luchthaven van bestemming*. Deze regel bevat slechts 1 versie die altijd geldig is. De regel levert *waar* op als zijnde *inconsistent* als de twee luchthavens aan elkaar gelijk zijn. We kunnen dit resultaat op de volgende manier in andere regels gebruiken:

- regelversie *Controleer of vlucht geen rondvlucht (altijd) is inconsistent*.
2. Een andere controle op de uitvoering van een regel is of een regel al dan niet is gevuurd (oftewel, of er aan de voorwaarden in het voorwaardendeel van de regel is voldaan). Om te controleren of een bepaalde regel gevuurd is gebruiken we het regelpredicaat *is gevuurd*, dat net als het regelpredicaat *is inconsistent* voorafgegaan moet worden door de desbetreffende regelversie.³¹

8.2 Tijdsafhankelijke aanvulling – check op volledige periode

De in de vorige paragraaf beschreven predicaten kunnen worden uitgebreid met een check of het betreffende predicaat gedurende een hele periode geldt. Deze check kan alleen worden uitgevoerd als het onderwerp van het predicaat tijdsafhankelijk is.

De uitbreiding houdt in dat “*gedurende de/het gehele <periode>*” wordt toegevoegd aan het predicaat.

Mogelijke perioden die hierbij kunnen worden gebruikt zijn:

- jaar – gedurende het gehele jaar,
- maand – gedurende de gehele maand.

N.B. Deze perioden komen overeen met perioden op de kalender. Zie voor de definitie paragraaf 3.8.

De toevoeging betekent dat de controle die het predicaat beschrijft alleen “*waar*” oplevert als op elk moment in de opgegeven periode de controle “*waar*” als resultaat heeft.

De uitbreiding van de predicaten met de check op volledige tijdseenheid verschilt taalkundig voor de predicaten in vragende en stellende vorm.

Vragende vorm.

De check op volledige tijdseenheid wordt vóór het predicaat geplaatst.

De vragende vorm van de kenmerkcheck wordt daarnaast ook aangepast door het werkwoord naar achteren te plaatsen conform de andere predicaten in de vragende vorm.

De check kan ook in ontkennende vorm worden gebruikt.

Voorbeelden:

³⁰ Deze regel staat verder uitgewerkt in paragraaf 9.5.

³¹ Aangezien deze constructie nagenoeg zelden wordt gebruikt en genomineerd staat uit RegelSpraak te halen, is het opnemen van een voorbeeld hierbij in dit document achterwege gelaten.

- gedurende het gehele jaar gelijk is aan
- gedurende de gehele maand leeg is
- gedurende het gehele jaar een <kenmerknaam> is / heeft
- niet gedurende het gehele jaar gelijk is aan

Stellende vorm.

De check op volledige tijdseenheid wordt ná het werkwoord ingevoegd in het predicaat.

Voorbeelden:

- is gedurende het gehele jaar gelijk aan
- is gedurende de gehele maand leeg
- is / heeft gedurende het gehele jaar een <kenmerknaam>
- is niet gedurende het gehele jaar gelijk aan

8.3 Enkele en samengestelde conditie

Als er slechts één specifiek onderdeel gecontroleerd moet worden, dan spreken we van een enkele conditie. Als er meerdere zaken gecontroleerd moeten worden, dan spreken we van een samengestelde conditie. Om de leesbaarheid en uitdrukking in de natuurlijke taal te bevorderen worden de gebruikte predicaten in beide gevallen afwijkend van elkaar genoteerd en moet de regel op een andere manier worden opgesteld. In de navolgende subparagrafen wordt dit verder besproken.

8.3.1 Enkele conditie

Bij een **enkele conditie** is er slechts één predicaat. Hiervoor wordt de vragende vorm van het predicaat gebruikt. Als de conditie wordt gebruikt als voorwaarde, dan wordt het woord *indien* voor de onderwerpexpressie gezet. Als de conditie gebruikt wordt als criterium, dan wordt het woord *moet* tussen de onderwerpexpressie en het predicaat gezet, en wordt de meervoudsvorm (van de vragende vorm) van het predicaat gebruikt.

Stel dat we willen controleren of een *Natuurlijk persoon* minderjarig is. Dit is het geval als de leeftijd van deze natuurlijke persoon kleiner is dan de leeftijd waarop mensen als volwassen worden beschouwd. We specificeren daarvoor een parameter *volwassenleeftijd*.

Parameter *volwassenleeftijd* : Numeriek (niet-negatief geheel getal) met eenheid jr

Met deze parameter kunnen we het predicaat *kleiner dan* gebruiken, dat controleert of een bepaalde waarde kleiner is dan een andere waarde. Als we onze controle willen gebruiken als criterium, dus in een consistentieregel, dan ziet een dergelijke regel er als volgt uit:

Regel controleer minderjarig
 geldig altijd
 De *leeftijd* van een *Natuurlijk persoon* moet kleiner zijn dan de *volwassenleeftijd*.

Als we onze controle willen gebruiken als voorwaarde, dus in het voorwaardendeel van een regel, dan ziet zo'n regel er als volgt uit:

Regel bepaal minderjarig
 geldig altijd
 Een *Natuurlijk persoon* is *minderjarig*
 indien zijn *leeftijd* kleiner is dan de *volwassenleeftijd*.

Syntax enkele conditie specificatie:

Opmerking 1: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

Opmerking 2: dit is slechts een deel van de totale syntax. De totale syntax is terug te vinden in hoofdstuk 13.

```

<voorwaardevergelijking> ::= <getalvergelijking> | <objectvergelijking> |
<tekstvergelijking> | <datumvergelijking> | <booleanvergelijking> |
<periodevergelijking>

<getalvergelijking> ::= <eenzijdigegetalvergelijking> | <tweezijdigegetalvergelijking>

<eenzijdigegetalvergelijking> ::= <getalexpressie>
<eenzijdigegetalvergelijkingsoperator>

<tweezijdigegetalvergelijking> ::= <getalexpressie>
<tweezijdigegetalvergelijkingsoperator> <getalexpressie>

<eenzijdigegetalvergelijkingsoperator> ::=
<eenzijdigegetalvergelijkingsoperatorenkelvoud> |
<eenzijdigegetalvergelijkingsoperatormeervoud>

<eenzijdigegetalvergelijkingsoperatorenkelvoud> ::= ("is" ("leeg" | "gevuld")) |
"voldoet aan de elfproef"

<eenzijdigegetalvergelijkingsoperatormeervoud> ::= ("zijn" ("leeg" | "gevuld")) |
"voldoen aan de elfproef"

<tweezijdigegetalvergelijkingsoperator> ::=
<tweezijdigegetalvergelijkingsoperatorenkelvoud> |
<tweezijdigegetalvergelijkingsoperatormeervoud>

<tweezijdigegetalvergelijkingsoperatorenkelvoud> ::= "is" ("gelijk aan" | "ongelijk aan"
| "groter dan" | "groter of gelijk aan" | "kleiner of gelijk aan" | "kleiner dan")

<tweezijdigegetalvergelijkingsoperatormeervoud> ::= "zijn" ("gelijk aan" | "ongelijk
aan" | "groter dan" | "groter of gelijk aan" | "kleiner of gelijk aan" | "kleiner dan")

```

8.3.2 Samengestelde conditie

Een **samengestelde conditie** bestaat uit een lijst met meerdere condities. De lijst met condities wordt gespecificeerd door gebruik te maken van bullets ('•'), oftewel elke conditie wordt voorafgegaan door een bullet. Verder moet gespecificeerd worden hoeveel van deze condities *waar* moeten zijn om de samengestelde conditie *waar* te laten zijn. Dit doen we door middel van het toevoegen van een **kwantificatie**. De volgende kwantificaties kunnen hierbij gebruikt worden:

- *alle* – aan alle condities in de lijst moet voldaan zijn om aan de samengestelde conditie te voldoen;
- *geen van de* – aan geen van de condities in de lijst mag voldaan zijn om aan de samengestelde conditie te voldoen;
- *ten minste* (één, twee, drie, vier of meer (genoteerd als getal)) *van de* – aan ten minste het gespecificeerde aantal condities in de lijst moet voldaan zijn om aan de samengestelde conditie te voldoen;
- *ten hoogste* (één, twee, drie, vier of meer (genoteerd als getal)) *van de* – aan hoogstens het gespecificeerde aantal condities in de lijst mag voldaan zijn om aan de samengestelde conditie te voldoen;
- *precies* (één, twee, drie, vier of meer (genoteerd als getal)) *van de* – aan precies het gespecificeerde getal aan aantal condities in de lijst moet voldaan zijn om aan de samengestelde conditie te voldoen.

Deze kwantificatie wordt op de volgende manier gebruikt bij een samengestelde conditie:

1. Bij een *samengestelde voorwaarde* moet de lijst met condities voorafgegaan worden door de woorden indien er aan <kwantificatie> volgende voorwaarden wordt voldaan.
2. Bij een *samengestelde voorwaarde* waarbij een attribuut, kenmerk of rol van slechts één instantie gecontroleerd wordt kan de onderwerpexpressie die naar die instantie verwijst opgenomen worden in de voorwaarde. De lijst zal dan voorafgegaan worden door de woorden indien <onderwerp expressie> aan <kwantificatie> volgende voorwaarden voldoet:.
3. Bij een *samengesteld criterium* moet de lijst voorafgegaan worden door de woorden er moet voldaan worden aan <kwantificatie> volgende criteria.
4. Bij een *samengesteld criterium* waarbij een attribuut, kenmerk of rol van slechts één instantie gecontroleerd wordt kan de onderwerpexpressie die naar die instantie verwijst opgenomen worden. De lijst zal dan voorafgegaan worden door de woorden <onderwerp expressie> moet voldoen aan <kwantificatie> volgende criteria.

Let op: voor de condities in de lijst wordt de stellende vorm van het predicaat gebruikt.

Voorbeeld: als we willen controleren of een *passagier* recht heeft op duurzaamheidskorting, dan moeten we twee punten nagaan. Ten eerste of de *reis* het kenmerk *duurzaam* heeft, en ten tweede of de *afstand tot bestemming* groot genoeg is om voor de korting in aanmerking te komen. Dit is een samengestelde conditie, bestaande uit twee condities die allebei *waar* moeten zijn om de samengestelde conditie *waar* te maken. Dit zouden we als volgt kunnen opschrijven:

Regel recht op duurzaamheidskorting
 geldig altijd
 Een *passagier* heeft *recht op duurzaamheidskorting*
 indien er aan alle van de volgende voorwaarden wordt voldaan:

- zijn *reis* is *duurzaam*
- de *afstand tot bestemming* van zijn *reis* is groter of gelijk aan *duurzaamheidskorting minimale afstand*.

Aangezien we hier een kenmerk en een attribuut van slechts één instantie controleren, kunnen we de deze regel ook als volgt opschrijven:

Regel recht op duurzaamheidskorting
 geldig altijd
 Een *passagier* heeft *recht op duurzaamheidskorting*
 indien hij aan alle volgende voorwaarden voldoet:

- zijn *reis* is *duurzaam*
- de *afstand tot bestemming* van zijn *reis* is groter of gelijk aan *duurzaamheidskorting minimale afstand*.

We zouden deze samengestelde conditie ook kunnen gebruiken als criterium in een consistentieregel. De regel hieronder zal *inconsistent* zijn wanneer de *Passagier* niet aan het samengestelde criterium voldoen:

Regel controleer duurzaamheidskorting
 geldig altijd
 Een *passagier* moet voldoen aan alle van de volgende criteria:

- zijn *reis* is *duurzaam*
- de *afstand tot bestemming* van zijn *reis* is groter of gelijk aan *duurzaamheidskorting minimale afstand*.

De lijst met condities kan bestaan uit enkele condities zoals hierboven besproken, maar kan ook weer zelf samengestelde condities bevatten. Zo kunnen samengestelde condities in elkaar genest worden waarbij de nestingsdiepte in principe onbeperkt is. Voor ieder nestingsniveau krijgt de voorwaarde één bullet ('•') erbij, het aantal bullets zal dus gelijk zijn aan het nestingsniveau. Het tweede niveau krijgt dan ook twee bullets, het derde drie etc.

Om het gebruik van geneste predicaten te illustreren breiden we het voorwaardendeel uit het bovenstaand voorbeeld uit, door toe te voegen dat een persoon ouder dan 64 jaar moet zijn om in aanmerking te komen voor duurzaamheidskorting. Gebruikmakend van enkele kenmerken van *Natuurlijk persoon* kunnen we dan het volgende schrijven:

Regel recht op duurzaamheidskorting voor ouderen
 geldig altijd
 Een *passagier* heeft *recht op duurzaamheidskorting*
 indien hij aan alle volgende voorwaarden voldoet:

- zijn *reis* is *duurzaam*
- de *afstand tot bestemming* van zijn *reis* is groter of gelijk aan *duurzaamheidskorting minimale afstand*
- hij voldoet aan geen van de volgende voorwaarden:
 - hij is een *passagier jonger dan 18 jaar*
 - hij is een *passagier van 18 tot en met 24 jaar*
 - hij is een *passagier van 24 tot en met 64 jaar*.

Syntax samengestelde conditie specificatie:

Opmerking 1: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.
 Opmerking 2: dit is slechts een deel van de totale syntax. De totale syntax is terug te vinden in hoofdstuk 13.

```
<genestesamengesteldevoorwaarde> ::= (<objectexpressie> | <referentie> | <aggregatie> |
"er") ("voldoet" | "voldoen" | "wordt voldaan") "aan" <voorwaardekwantificatie>
"volgende voorwaarde"["n"] ":" <samengesteldevoorwaardeonderdeel>
```

```
<voorwaardekwantificatie> ::= "de" | "alle" | "geen van de" | ("ten minste" | "ten
hoogste" | "precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")
```

```
<samengesteldcriterium> ::= "er wordt voldaan aan" ("het volgende criterium:" |
(<consistentiekwantificatie> "volgende criteria:") <samengesteldcriteriumonderdeel>
```

```
<consistentiekwantificatie> ::= "alle" | "geen van de" | ("ten minste" | "ten hoogste"
| "precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")
```

8.4 Tijdsafhankelijke conditie

8.4.1 Algemeen

Een tijdsafhankelijke conditie betreft een conditie waarbij het resultaat van de conditie tijdsafhankelijk is. Dat wil zeggen dat de conditie waar of onwaar kan zijn op verschillende momenten in de tijd.

Alle condities kunnen zowel tijdsafhankelijk als niet-tijdsafhankelijk zijn. Of de conditie tijdsafhankelijk of niet-tijdsafhankelijk is hangt er vanaf of de in de expressie gebruikte gegevens tijdsafhankelijk zijn. Als één van de gegevens tijdsafhankelijk is, dan zijn de volledige expressie en conditie tijdsafhankelijk.

Daarnaast is er één specifieke tijdsafhankelijke conditie. Deze conditie betreft de verwijzing naar een specifieke periode op basis van datum(s) (zie paragraaf 8.4.3).

Het gebruik van een tijdsafhankelijke conditie in het voorwaardendeel van de regel is beschreven in paragraaf 10.3.

Tijdsafhankelijke condities kunnen ook worden gebruikt bij twee specifieke tijdsafhankelijke expressies, nl. “het totaal van” (zie paragraaf 7.1) en “het tijdsevenredig deel” (zie paragraaf 7.3.2).

Let op:

Bij gebruik van een tijdsafhankelijke conditie bij een expressie moet de expressie tussen haakjes worden geplaatst of als variabele in de regel worden opgenomen.

De reden hiervoor is dat de regel anders ambigu wordt doordat niet duidelijk is of de tijdsafhankelijke conditie alleen de expressie betreft of een voorwaarde is.

8.4.2 Gedurende de tijd dat

Als een tijdsafhankelijke conditie wordt gebruikt dan wordt die conditie vooraf gegaan door de uitdrukking “gedurende de tijd dat...”. Bijvoorbeeld bij een enkelvoudige conditie:

gedurende de tijd dat **hij** een **recht op belastingvermindering** heeft

In dit voorbeeld is het bezittelijk kenmerk **recht op belastingvermindering** een tijdsafhankelijk kenmerk dat het object op bepaalde momenten in de tijd wel heeft en op andere momenten niet. De conditie is waar in de periode(s) waarin het object wel het kenmerk heeft.

In een samengestelde conditie ziet het gebruik van *gedurende de tijd dat* er bijvoorbeeld zo uit:

gedurende de tijd dat er aan alle volgende voorwaarden wordt voldaan

- **hij** heeft een **recht op belastingvermindering**
- zijn **geboortedatum** is eerder dan **1-1-2002**

In dit voorbeeld is de eerste conditie tijdsafhankelijk, maar de tweede niet (de geboortedatum heeft geen verschillende waarden op verschillende momenten). De samengestelde conditie is als geheel wel tijdsafhankelijk en wordt voorafgegaan door *gedurende de tijd dat*.

De uitzondering op het gebruik van *gedurende de tijd dat* vormt de situatie waarbij de specifieke tijdsafhankelijke conditie (zie paragraaf 8.4.3) als enkelvoudige conditie wordt toegepast. In dat geval wordt alleen de conditie gebruikt zonder daaraan voorafgaand de uitdrukking *gedurende de tijd dat*.

8.4.3 vanaf / van...tot / van...tot en met / tot / tot en met

Een specifieke tijdsafhankelijke conditie is de conditie waarbij een specifieke periode op basis van datums wordt gespecificeerd. Die periode kan worden uitgedrukt met:

- vanaf ...
- van ... tot ...
- van ... tot en met ...
- tot ...
- tot en met ...

De precieze betekenis van deze periode aanduiding is toegelicht in paragraaf **Fout!**
Verwijzingsbron niet gevonden..

Voorbeeld van de toepassing van deze conditie in een regel met een enkelvoudige conditie in het voorwaardendeel:

De te betalen belasting van een passagier moet berekend worden als zijn belasting op basis van afstand plus zijn belasting op basis van reisduur van dd. 1-1-2024 tot dd. 8-2-2025

In dit voorbeeld zijn de datums als literal opgenomen (“harde” waarde), maar in plaats daarvan kunnen alle andere expressies die als resultaat een datum hebben worden gebruikt (attributen, parameters, berekening van datum).

Als deze conditie wordt opgenomen in een samengestelde conditie, dan wordt dit uitgedrukt met *het is de periode* gevolgd door de conditie. Bijvoorbeeld:

De te betalen belasting van een passagier moet berekend worden als zijn belasting op basis van afstand plus zijn belasting op basis van reisduur gedurende de tijd dat er aan alle volgende voorwaarden wordt voldaan

- hij heeft een recht op belastingvermindering
- het is de periode van dd. 1-1-2024 tot dd. 8-2-2025

9 Resultaatdeel

Het resultaatdeel van een regel beschrijft een actie: datgene wat er gebeurt als de regel wordt toegepast. Dit deel is verplicht, oftewel het is niet mogelijk om een RegelSpraak-regel te schrijven zonder een resultaatdeel. In welke (reken)volgorde het resultaatdeel van een RegelSpraak-regel toegepast wordt in relatie tot het voorwaardendeel en variabelendeel staat beschreven in paragraaf 11.1.

De volgende resultaatdelen zijn hierbij als actie mogelijk welke in de genoemde subparagrafen verder besproken zullen worden:

1. Gelijktelling (zie paragraaf 9.1)
2. Kenmerktoekenning (zie paragraaf 9.2)
3. ObjectCreatie (zie paragraaf 9.3)
4. FeitCreatie (zie paragraaf 9.4)
5. Consistentieregels (zie paragraaf 9.5)
6. Initialisatie (zie paragraaf 9.6)
7. Verdeling (zie paragraaf 9.7)
8. Dagsoortdefinitie (zie paragraaf 9.8)

Syntax resultaatdeel specificatie:

```
<resultaatdeel> ::= <gelijktelling> | <kenmerktoekenning> | <objectcreatie> |  
<feitcreatie> | <consistentieregel> | <initialisatie> | <verdeling> |  
<dagsoortdefinitie>
```

9.1 Gelijktelling

Een **gelijktelling** is een actie waarbij een attribuut een waarde krijgt die wordt uitgedrukt door een expressie. Hierbij is het taalpatroon afhankelijk van het feit of hier sprake is van een berekeningsexpressie.

- 1) Bij het toekennen van een waarde uit een berekeningsexpressie is het taalpatroon “De/het <attribuut> van een <onderwerpexpressie> moet berekend worden als <expressie>” waarbij de expressie zowel een getalexpressie als datumexpressie kan zijn (met beide kan immers gerekend worden). Een voorbeeld uit de TOKA-casus is de *te betalen belasting*, in het geval dat de vlucht geen rondvlucht is en de vlucht niet duurzaam is:

Regel Te betalen belasting
 geldig altijd
 De *te betalen belasting* van een *Natuurlijk persoon* moet berekend worden als
 zijn *belasting op basis van afstand* plus zijn *belasting op basis van reisduur*.

- 2) Bij het toekennen van een andere waarde is het taalpatroon “De/het <attribuut> van een <onderwerpexpressie> moet gesteld worden op <waarde>”. Als we bovenstaande regel schrijven met een variabele (zie paragraaf 4.3 en hoofdstuk 11), dan ziet deze er als volgt uit:

Regel Te betalen belasting
 geldig altijd
 De *te betalen belasting* van een *Natuurlijk persoon* moet gesteld worden op zijn
belasting op basis van afstand.

De linker- en de rechterkant van een gelijkstelling moeten hetzelfde datatype hebben, en indien er verschillende eenheden worden gebruikt dan moeten deze in elkaar om te rekenen zijn via een eenheidssysteem (zie paragraaf 3.7).

Wanneer gelijkgesteld wordt aan een leegwaarde zal het resultaat ook een leegwaarde zijn.

Syntax gelijkstelling specificatie:

```
<gelijkstelling> ::= (<gelijkstellingtoekenning> | <gelijkstellingberekening>)

<gelijkstellingtoekenning> ::= <attribuutvanonderwerp> "moet gesteld worden op"
<expressie>

<gelijkstellingberekening> ::= <attribuutvanonderwerp> "moet berekend worden als"
(<getalexpressie> | <datumexpressie>)
```

9.2 Kenmerktoekenning

Een kenmerktoekenning bepaalt of een instantie van een objecttype het kenmerk moet krijgen en kent het kenmerk toe als dat het geval is. Bij een kenmerktoekenning wordt normaal gesproken altijd een voorwaardendeel (zie paragraaf 4.3 en hoofdstuk 9.9) gespecificeerd dat beschrijft wanneer de instantie voldoet aan het kenmerk. Het weglaten van een voorwaarde zorgt ervoor dat het kenmerk altijd van toepassing is en dat is vaak niet nuttig, vandaar dat in dit soort situaties een voorwaardendeel doorgaans wordt gespecificeerd.

Als een kenmerk niet bezittelijk of bijvoeglijk is, is het taalpatroon "Een <objecttype> is een <kenmerk>". In RegelSpraak ziet dit er als volgt uit:

Een vlucht is een belaste reis

Indien een kenmerk echter als bijvoeglijk is gespecificeerd (zie paragraaf 3.5 voor meer informatie hierover) dan leidt dat tot een afwijkend taalpatroon: "Een <objecttype> is <kenmerk>". Dit ziet er als volgt uit in RegelSpraak:

Een vlucht is duurzaam

Indien een kenmerk tot slot als bezittelijk is gespecificeerd (zie paragraaf 3.5 voor meer informatie hierover) dan leidt dat tot een afwijkend taalpatroon: "Een <objecttype> heeft <kenmerk>". In RegelSpraak ziet dit er als volgt uit:

Een passagier heeft recht op korting

Syntax kenmerktoekenning specificatie:

```
<kenmerktoekenning> ::= <onderwerpketen> ("is" | "heeft") ["een"] <kenmerknaam>
```

Belangrijk om hierbij op te merken is dat kenmerken onwaar zijn als er geen kenmerktoekenning is gedaan. Om die reden kan er ook geen negatieve kenmerktoekenning worden gedaan. Dus "Een vlucht is niet duurzaam" is in RegelSpraak niet toegestaan.

9.3 ObjectCreatie

Instanties van objecttypen worden voor het uitvoeren van RegelSpraak-regels automatisch gecreëerd op basis van de invoergegevens. In de TOKA casus bijvoorbeeld zullen er instanties gecreëerd worden van *Vlucht* en *Natuurlijk persoon* aan de hand van de vluchten en natuurlijke personen in de echte wereld waar de wetgeving op toegepast moet worden. Hoe deze instanties precies hun waardes krijgen vanuit de invoer valt buiten de scope van dit document.

Soms is het nodig om extra instanties aan te maken die niet gebaseerd zijn op de invoer. Het

objecttype *Contingent treinmiles* bijvoorbeeld wordt gebruikt om het aantal toe te kennen *treinmiles* te bepalen. Instanties van dit objecttype kunnen niet vanuit de invoer gecreëerd worden. Immers, de *treinmiles* zijn bij de invoer nog niet bekend omdat ze pas bepaald worden bij het uitvoeren van de regels. Het is in RegelSpraak dan ook mogelijk om, tijdens het uitvoeren van een regel, een nieuwe instantie van een objecttype te creëren. Dit is de actie **ObjectCreatie**.

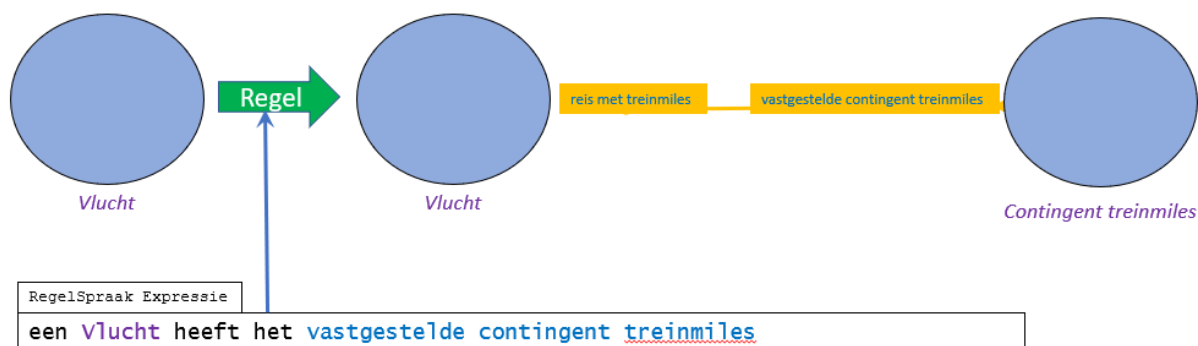
Een nieuwe instantie kan bij ObjectCreatie alleen gecreëerd worden in relatie met een al bestaande instantie. Deze relatie zal moeten zijn gespecificeerd met een feittype. Als we bijvoorbeeld een instantie van *Contingent treinmiles* willen creëren, dan is dat alleen zinvol als we al een instantie van *Vlucht* hebben waar we de instantie aan kunnen koppelen. Immers, de instantie zal gebruikt worden voor het bepalen van de *treinmiles* van de vlucht. Behalve het specificeren van *Contingent treinmiles* zal er dus ook een feittype gespecificeerd moeten worden, dat het objecttype verbindt met objecttype *Vlucht*.

Objecttype het *Contingent treinmiles*
 het *totaal aantal treinmiles* Numeriek (positief geheel getal);
 het *aantal treinmiles op basis van aantal passagiers* Numeriek (positief geheel getal);
 het *restant na verdeling* Numeriek (positief geheel getal);

Feittype reis met contingent treinmiles
 de *reis met treinmiles* *Vlucht*
 het *vastgestelde contingent treinmiles* *Contingent treinmiles*
 één *reis met treinmiles* heeft één *vastgestelde contingent treinmiles*;

De actie om een nieuw object te creëren heeft het taalpatroon “Een <onderwerpexpressie> heeft <ro1>”. Een ObjectCreatie actie om een instantie van *Contingent treinmiles* te creëren en die met een nieuw feit van *reis met contingent treinmiles* te koppelen aan een instantie van *Vlucht* ziet er als volgt uit:

een *Vlucht* heeft het *vastgestelde contingent treinmiles*



Figuur 16: een ObjectCreatie expressie. Een nieuwe instantie van *Contingent treinmiles* wordt aangemaakt die met een nieuw feit *reis met contingent treinmiles* verbonden is met de al bestaande instantie van *Vlucht*.

Het is ook mogelijk om attributen van de nieuwe instantie in dezelfde expressie meteen een waarde te geven. Het taalpatroon wordt dan “met <attribuut1 van nieuwe instantie> gelijk aan <expressie1> en <attribuut2 van nieuwe instantie> gelijk aan <expressie2>”. We kunnen bijvoorbeeld het attribuut *aantal treinmiles op basis van aantal passagiers* van onze nieuwe instantie van *Contingent treinmiles* meteen bij ObjectCreatie de juiste waarde toekennen. We gebruiken hiervoor een parameter *aantal treinmiles per passagier voor contingent* die aangeeft hoeveel *treinmiles* er per passagier moeten worden toegekend.

Parameter de *aantal treinmiles per passagier voor contingent*: Numeriek (positief geheel getal)

Met deze parameter en de ObjectCreatie actie kunnen we nu de volgende regel schrijven:

Regel vastgestelde contingent treinmiles

geldig altijd

Een *Vlucht* heeft het *vastgestelde contingent treinmiles* met *aantal treinmiles op basis van aantal passagiers* gelijk aan het *aantal passagiers* van de *Vlucht* maal het *aantal treinmiles per passagier voor contingent*.

Syntax ObjectCreatie specificatie:

```
<objectcreatie> ::= "Een" <onderwerpketen> "heeft een" <rolnaam> [ "met"
<waardetoekenning> [ ("," <waardetoekenning>)* "en" <waardetoekenning>] ]

<waardetoekenning> ::= <attribuutwaardetoekenning> | <kenmerkwaardetoekenning>

<attribuutwaardetoekenning> ::= <attribuut> "gelijk aan" <expressie>

<kenmerkwaardetoekenning> ::= <kenmerknaam> "gelijk aan" ("waar" | "onwaar")
```

9.4 FeitCreatie

In de vorige paragraaf hebben we een nieuwe instantie van *Contingent treinmiles* gemaakt, dat met een feit was verbonden aan een instantie van *Vlucht*. Met deze nieuwe instantie kunnen we nu het totaal aantal treinmiles bepalen dat bij een bepaalde vlucht hoort. Deze treinmiles moeten vervolgens verdeeld worden over de passagiers van de vlucht. Om dat te doen, moeten we de nieuwe instantie verbinden met de instanties van *Natuurlijk persoon* die de rol *passagier* invullen bij een instantie van *Vlucht*; we moeten een nieuw feit creëren. Hiervoor moeten we eerst een feittype specificeren dat deze twee objecttypen met elkaar in verband brengt. Nieuwe instanties van een feittype kunnen aangemaakt worden met een **FeitCreatie** actie.

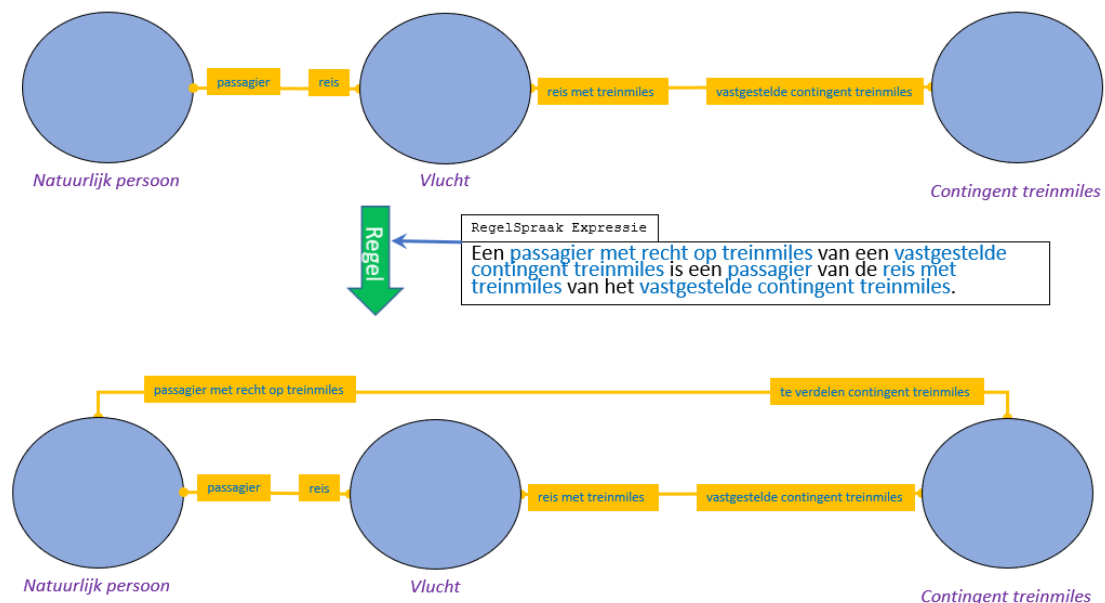
Feittype verdeling contingent treinmiles over passagiers
 het *te verdelen contingent treinmiles* *Contingent Treinmiles*
 de *passagier met recht op treinmiles* *Natuurlijk persoon*
 één *te verdelen contingent treinmiles* wordt verdeeld over meerdere *passagiers met recht op treinmiles*

We kunnen nu een regel schrijven die ons nieuwe feit aanmaakt:

Regel passagier met recht op treinmiles

geldig altijd

Een *passagier met recht op treinmiles* van een *vastgestelde contingent treinmiles* is een *passagier* van de *reis met treinmiles* van het *vastgestelde contingent treinmiles*.



Figuur 17: een FeitCreatie expressie. Een instantie van *Contingent treinmiles* wordt verbonden met een instantie van *Natuurlijk persoon* via het feittype *verdeling contingent treinmiles over passagiers*.

Syntax FeitCreatie specificatie:

```
<feitcreatie> ::= "Een" <rolnaam> "van een" <onderwerpketen> "is" <bepaaldlidwoord>
<rolnaam> "van" <bepaaldlidwoord> <onderwerpketen>
```

9.5 Consistentieregels

Consistentieregels zijn regels die niet bedoeld zijn om aan een attribuut of een kenmerk een waarde te geven, maar om te controleren of de waarde van een attribuut, rol of instantie aan bepaalde criteria voldoet. Criteria voor consistentieregels zijn condities (zie hoofdstuk 7). Ze worden geschreven als een verplichting met het woord *moeten*. Als er niet aan de criteria wordt voldaan, dan zal een consistentieregel als resultaat *niet waar*, oftewel *inconsistent*, opleveren. Dit resultaat kan dan in andere regels gebruikt worden.

Als er sprake is van één enkel criterium, dan kan een consistentieregel opgesteld worden als een enkele conditie (zie paragraaf 8.3.1). Dat betekent dat een onderwerpexpressie gecombineerd moet worden met de enkele schrijfwijze voor een criterium van het predicaat. Een voorbeeld van een enkel criterium is de controle of een vlucht geen rondvlucht is. Dat kunnen we bepalen door te kijken naar de luchthavens waar het vliegtuig opstijgt en weer landt. Als dat dezelfde luchthavens zijn, dan is de vlucht een rondvlucht. In RegelSpraak betekent dat, dat we de twee attributen *luchthaven van vertrek* en *luchthaven van bestemming* van *Vlucht* met elkaar moeten vergelijken. We gebruiken hiervoor het predicaat *moet ongelijk zijn aan*, met de schrijfwijze die voor een enkel criterium gebruikt moet worden (zie paragraaf 8.3.1). Dit leidt tot de volgende regel:

Regel Controleer of vlucht geen rondvlucht is
 geldig altijd
 De *luchthaven van vertrek* van een *vlucht* moet ongelijk zijn aan de *luchthaven van bestemming* van de *vlucht*.

Als er sprake is van meerdere criteria, dan moet de regel anders geschreven worden zoals besproken in paragraaf 8.3.2. De criteria vormen dan een lijst met bullets die voorafgegaan moeten worden door de woorden *moet voldoen aan <kwantificatie> volgende criteria*:. Tevens moet een andere schrijfwijze van het predicaat gebruikt worden.

Een voorbeeld van samengestelde criteria is het controleren of de TOKA wet van toepassing is op een *Vlucht*. Het eerste criterium hiervoor is of een *Vlucht belast* is. Dit kunnen we controleren met een kenmerkcheck. Het tweede criterium is of een *Vlucht passagiers* heeft. Dit kunnen we controleren met een vergelijkingspredicaat en het attribuut *aantal passagiers*. Als niet aan één van deze criteria voldaan wordt, dan zal de regel *inconsistent* opleveren. Dit leidt tot de volgende regel:

Regel Controleer of de TOKA wet van toepassing is
 geldig altijd
 Een *reis* moet voldoen aan ten minste één van de volgende criteria:

- de *reis* is *belast*
- het *aantal passagiers* van de *reis* is groter dan 0.

Syntax consistentieregel specificatie

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```

<consistentieregel> ::= <enkelvoudigeconsistentieregel | <toplevelsamengesteldcriterium>
| <uniciteitscontrole>

<enkelvoudigeconsistentieregel> ::= <getalconsistentie> | <datumconsistentie> |
<tekstconsistentie> | <objectconsistentie>

<getalconsistentie> ::= <getalexpressie> "moet"
(<topleveleenzijdigegetalvergelijkingsoperatormeervoud> |
<topleveltweezijdigegetalvergelijkingsoperatormeervoud>)

<datumconsistentie> ::= <datumexpressie> "moet"
(<topleveleenzijdigedatumvergelijkingsoperatormeervoud> |
<topleveltweezijdigedatumvergelijkingsoperatormeervoud>)

<tekstconsistentie> ::= <tekstexpressie> "moet"
(<topleveleenzijdigetekstvergelijkingsoperatormeervoud> |
topleveltweezijdigetekstvergelijkingsoperatormeervoud)

<objectconsistentie> ::= <objectexpressie> "moet"
(<topleveleenzijdigeobjectvergelijkingsoperatormeervoud> |
<topleveltweezijdigeobjectvergelijkingsoperatormeervoud>)

<toplevelsamengesteldcriterium> ::= "er moet worden voldaan aan" ("het volgende
criterium:" | (<consistentiekwantificatie> "volgende criteria:")
<samengesteldcriteriumonderdeel>

<samengesteldcriteriumonderdeel> ::= \n \t <genestcriterium> (\n <genestcriterium>)+

<genestcriterium> ::= ("•")+ (<voorwaardevergelijking> | <samengesteldcriterium>)

<samengesteldcriterium> ::= "er wordt voldaan aan" ("het volgende criterium:" |
(<consistentiekwantificatie> "volgende criteria:") <samengesteldcriteriumonderdeel>

<uniciteitscontrole> ::= (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>)
<vereniging>* "moeten uniek zijn."

<vereniging> ::= "verenigd met" (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>)

<alleattribuutvanonderwerp> ::= "de" <meervoudsvorm> "van alle" (objecttypenaam |
<rolnaam> "van" <onderwerpketen>)] ["van" <onderwerpketen>]

<uniciteitconcatenatie> ::= "de concatenatie van" <meervoudsvorm> ("," <meervoudsvorm>)*
"en" <meervoudsvorm> "van alle" (objecttypenaam | <rolnaam> "van" <onderwerpketen>)]
["van" <onderwerpketen>]

```

9.6 Initialisatie

Initialisatie wordt gebruikt voor attributen waarvan de waarde niet leeg mag zijn. Een initialisatieregel controleert of een attribuut een waarde bevat, welke eventueel nog wordt toegekend vanuit een specifieke regel zoals een gelijkstelling (zie paragraaf 9.1) of een verdeling (zie paragraaf 9.7). Als dit niet het geval is, dan kent de initialisatieregel een gespecificeerde waarde toe aan het attribuut. Het taalpatroon dat hiervoor gebruikt wordt, is: "De/het <attribuut> van een <onderwerpexpressie> moet geïnitieerd worden op <waarde>". Een voorbeeld in RegelSpraak kan er dan als volgt uitzien:

Parameter de *initiele belasting* : Bedrag

Regel Initialiseer te betalen belasting op initiele belasting

geldig altijd

De *te betalen belasting* van een *passagier* moet geïnitieerd worden op de *initiele belasting*.

Syntax initialisatie specificatie:

```
<initialisatie> ::= <attribuutvanonderwerp> "moet geïnitieerd worden op"
<expressie>.
```

9.7 Verdeling

Een **verdeling** wordt gebruikt om een waarde van een numeriek attribuut (ook wel het **verdeelp plafond** genoemd) te verdelen over attribuutwaarden van andere objecten (ook wel het **aandeel in een verdeelp plafond** genoemd) waartussen een meervoudig feittype is gedefinieerd. In een verdeling wordt gesproken over een **verdeler** en een **ontvanger**. De verdeler is de rol met het attribuut waarvan de waarde (het verdeelp plafond) wordt verdeeld en de ontvanger is de rol met het attribuut waaraan een deel van de waarde (het aandeel in het verdeelp plafond) wordt toegekend. In de TOKA-casus moet bijvoorbeeld bij een reis een aantal treinmiles (verdeler) verdeeld worden over de passagiers van de reis (ontvanger).

9.7.1 Verdeling over alle ontvangers

Bij een verdeling wordt altijd vastgelegd of er gelijk verdeeld moet worden of naar rato van de waarde van een ander attribuut waarbij dit altijd een attribuut van de ontvanger is. Wanneer naar rato wordt verdeeld krijgen de instanties van de rol met een hogere waarde voor dit 'naar rato attribuut' een groter aandeel van het te verdelen bedrag dan instanties met een kleinere waarde.

Twee basisverdelingen staan hieronder als voorbeeld waarvan het eerste voorbeeld een gelijke verdeling illustreert en het tweede voorbeeld een verdeling naar rato (op basis van woonregio factor):

Regel Verdeling treinmiles in gelijke delen
geldig altijd

Het **totaal aantal treinmiles** van een **te verdelen contingent treinmiles** wordt verdeeld over de **treinmiles** van alle **passagiers met recht op treinmiles** van het **te verdelen contingent treinmiles**, waarbij wordt verdeeld in gelijke delen.

Regel Verdeling treinmiles op basis van woonregio factor
geldig altijd

Het **totaal aantal treinmiles** van een **te verdelen contingent treinmiles** wordt verdeeld over de **treinmiles** van alle **passagiers met recht op treinmiles** van het **te verdelen contingent treinmiles**, waarbij wordt verdeeld naar rato van de **woonregio factor**.

Ter illustratie van de twee bovenstaande regels zijn hieronder rekenvoorbeelden toegevoegd. Wanneer bijvoorbeeld een contingent van 1000 treinmiles wordt verdeeld over twee passagiers, Tom met woonregio factor 3 en Maria met woonregio factor 2, dan worden deze treinmiles als volgt over hen verdeeld:

- Bij verdeling in gelijke delen krijgen Tom en Maria hetzelfde aandeel van het beschikbare contingent treinmiles. Oftewel, $1000 / 2 = 500$ treinmiles per passagier.
- Bij verdeling op basis van woonregio factor wordt eerst het aantal woonregio factoren bij elkaar opgeteld. Die van Tom is 3 en die van Maria 2, dus samen 5. Dat betekent dat Tom $3/5^e$ deel van de 1000 beschikbare treinmiles krijgt, dus 600 treinmiles. Maria krijgt $2/5^e$ deel, dus 400 treinmiles.

9.7.2 Verdeling over groepen ontvangers in een volgorde

Binnen een verdeling kan een groepering van de ontvangers gespecificeerd worden op basis van een attribuut van de ontvangers. Hierbij specificeer je ook altijd of je afnemend of toenemend wil verdelen op basis van dit attribuut. Met een verdeling over groepen kun je bijvoorbeeld passagiers met een lagere leeftijd een grotere hoeveelheid treinmiles toekennen dan passagiers

met een hogere leeftijd.

Zodra verdeeld wordt over groepen moet ook iets gespecificeerd worden over wat er gebeurt wanneer er meerdere instanties zijn die in dezelfde groep vallen (meerdere passagier met dezelfde leeftijd bijvoorbeeld). Hierbij bestaat opnieuw de mogelijkheid om dit in gelijke delen te doen of naar rato op basis van een attribuut van de ontvanger. Hieronder volgt een voorbeeld met een verdeling over groepen³²:

Regel Verdeling treinmiles op basis van leeftijd en woonregio factor
geldig altijd

Het **totaal aantal treinmiles** van een **te verdelen contingent treinmiles** wordt verdeeld over de **treinmiles** van alle **passagiers met recht op treinmiles** van het **te verdelen contingent treinmiles**, waarbij wordt verdeeld:

- op volgorde van toenemende de **leeftijd**,
- bij een even groot criterium naar rato van de **woonregio factor**.

Als onverdeelde rest blijft het **restant na verdeling** van het **te verdelen contingent treinmiles** over.

Ter illustratie van het bovenstaande voorbeeld volgt een rekenvoorbeeld. Er wordt een contingent van 1200 treinmiles verdeeld over 3 passagiers conform onderstaande tabel:

Passagier	Leeftijd	Woonregio factor
Wilma	20	3
Hans	20	1
Diederik	22	2

Tabel 17: rekenvoorbeeld bij Verdeling.

Het contingent wordt verdeeld op basis van toenemende leeftijd, dus eerst wordt berekend welk aandeel Wilma en Hans krijgen, gezien zij het jongst zijn. Aangezien zij even oud zijn, wordt hierbij rekening gehouden met de woonregio factor. De woonregio factoren van Wilma en Hans bij elkaar opgeteld zijn $3 + 1 = 4$, wat betekent dat Wilma $3/4^e$ deel van het te verdelen contingent (1200) krijgt oftewel 900, en Hans krijgt $1/4^e$ deel oftewel 300. Omdat eerst naar de jongste passagiers wordt gekeken en geen maximum per passagier is ingesteld, is het contingent treinmiles nu al volledig verdeeld. Dat betekent dat Diederik geen treinmiles krijgt en er geen restant na verdeling is.

9.7.3 Gebruik van een maximale aanspraak

Als naar rato wordt verdeeld kan bij beide bovengenoemde vormen van verdeling een zogenaamde **aanspraak** toegevoegd worden. Dit is niet mogelijk bij gelijke verdeling omdat een gelijke verdeling met een maximum aanspraak voor meerdere uitleg vatbaar is.

Deze aanspraak is altijd een numeriek attribuut van de ontvanger en beschrijft hoeveel een ontvanger maximaal kan krijgen uit de verdeling. Dit is handig om te gebruiken in combinatie met het verdelen over groepen want zonder maximum zou, zoals hiervoor aangetoond, het hele te verdelen bedrag over de eerste groep verdeeld worden.

Als een maximale aanspraak wordt gespecificeerd, dan moet ook altijd een onverdeelde rest worden gespecificeerd omdat door toepassing van het maximum het totale verdeelplafond mogelijk niet helemaal wordt toegewezen.

Hieronder staat een voorbeeld van een verdeling met een aanspraak:

³² Let op dat in het voorbeeld ook iets over een onverdeelde rest staat, dit is om het voorbeeld compleet en correct te houden. De onverdeelde rest wordt later in deze paragraaf verder toegelicht.

Regel Verdeling treinmiles op basis van woonregio factor en met maximum waarde geldig altijd

Het totaal aantal treinmiles van een te verdelen contingent treinmiles wordt verdeeld over de treinmiles van alle passagiers met recht op treinmiles van het te verdelen contingent treinmiles, waarbij wordt verdeeld:

- naar rato van de woonregio factor,
- met een maximum van het maximaal aantal te ontvangen treinmiles.

Als onverdeelde rest blijft het restant na verdeling van het te verdelen contingent treinmiles over.

Ter illustratie van het bovenstaande voorbeeld volgt wederom een rekenvoorbeeld. Hierbij gebruiken we gegevens die we al eerder hebben gebruikt: Er wordt een contingent van 1200 treinmiles verdeeld over 3 personen, zie Tabel 17. Daarnaast geldt een maximum van 300 treinmiles per passagier. Volgens deze regel is leeftijd niet relevant maar wordt wel verdeeld op basis van woonregio factor. Samen hebben Wilma, Hans en Diederik $3 + 1 + 2 = 6$ woonregio factoren. Dat betekent dat Wilma recht heeft op $3/6^e$ van de beschikbare 1200 treinmiles, dus 600 treinmiles. Dit mag echter niet volgens de tweede voorwaarde van deze regel. 600 treinmiles overschrijdt immers het maximum van 300 treinmiles per passagier. Wilma krijgt dus 300 treinmiles. Hans krijgt $1/6^e$ van de beschikbare 1200 treinmiles, dus 200 treinmiles. Diederik heeft recht op $2/6^e$ van de 1200 treinmiles, dus 400 treinmiles. Dit is wederom meer dan het maximum van 300, waardoor Diederik 300 treinmiles krijgt. Nu alle passagiers hun treinmiles hebben gekregen, zijn nog niet alle treinmiles verdeeld. Er zijn nog $1200 - 300 - 200 - 300 = 400$ treinmiles beschikbaar. Dit is het restant na verdeling.

9.7.4 Gebruik van afronding

Als laatste kan het te ontvangen aantal treinmiles ook afgerond worden. Dit kan in deze situatie alleen naar beneden maar er kan wel een aantal decimalen gespecificeerd worden.

Als wordt afgerond, dan moet ook altijd een onverdeelde rest worden gespecificeerd omdat door toepassing van de afronding het totale verdeelplafond mogelijk niet helemaal wordt toegewezen.

Hieronder staat een voorbeeld van een verdeling met een afronding:

Regel Verdeling treinmiles op basis van woonregio factor met afronding geldig altijd

Het totaal aantal treinmiles van een te verdelen contingent treinmiles wordt verdeeld over de treinmiles van alle passagiers met recht op treinmiles van het te verdelen contingent treinmiles, waarbij wordt verdeeld:

- naar rato van de woonregio factor,
- afgerond op 0 decimalen naar beneden.

Als onverdeelde rest blijft het restant na verdeling van het te verdelen contingent treinmiles over.

Ter illustratie van het bovenstaande voorbeeld volgt hier een rekenvoorbeeld. We verdelen 1001 treinmiles over 2 passagiers: Tom met woonregio factor 3 en Maria met woonregio factor 2. Hier wordt weer verdeeld op basis van woonregio factor, dus wordt eerst het aantal woonregio factoren bij elkaar opgeteld. Die van Tom is 3 en die van Maria 2, dus samen 5. Dat betekent dat Tom $3/5^e$ deel van de 1001 beschikbare treinmiles krijgt, dus 600,6 treinmiles. Maria krijgt $2/5^e$ deel, dus 400,4 treinmiles. Het aantal treinmiles dat een passagier ontvangt moet echter naar beneden worden afgerond op 0 decimalen. Daardoor krijgt Tom 600 treinmiles en Maria 400. Dat betekent dat na deze verdeling nog $1001 - 600 - 400 = 1$ treinmiles over blijft als restant na verdeling.

9.7.5 Gebruik van onverdeelde rest

In het geval van verdelen met een aanspraak of een afronding kan het voorkomen dat na het verdelen een restant waarde overblijft. Deze waarde moet worden vastgelegd in een attribuut van de verdeler.

Voor de volledigheid is hierna nog een voorbeeld opgenomen van een verdeling over groepen met aanspraak en afronding.

Regel Verdeling treinmiles op basis van leeftijd, woonregio factor, met maximum waarde en afronding

geldig altijd

Het totaal aantal treinmiles van een te verdelen contingent treinmiles wordt verdeeld over de treinmiles van alle passagiers met recht op treinmiles van het te verdelen contingent treinmiles, waarbij wordt verdeeld:

- op volgorde van toenemende de leeftijd,
- bij een even groot criterium naar rato van de woonregio factor,
- met een maximum van het maximaal aantal te ontvangen treinmiles,
- afgerond op 0 decimalen naar beneden.

Als onverdeelde rest blijft het restant na verdeling van het te verdelen contingent treinmiles over.

Ter illustratie van het bovenstaande voorbeeld volgt hier een rekenvoorbeeld. In dit rekenvoorbeeld worden 1800 treinmiles verdeeld over 5 mensen. De leeftijd, woonregio factor, maximaal te ontvangen treinmiles en de uiteindelijk ontvangen treinmiles staan weergegeven in onderstaande tabel. Onder de tabel is uitgelegd hoe de ontvangen treinmiles berekend zijn.

Naam	Leeftijd	Woonregio factor	Maximaal te ontvangen treinmiles	Ontvangen treinmiles
Piet	22	3	500	500
Jan	39	2	500	500
Klaas	39	1	500	433
Nel	66	3	250	250
Ria	66	1	250	91

Tabel 18: gegevens voor rekenvoorbeeld verdelingen.

Het eerste criterium dat belangrijk is bij het uitwerken van dit voorbeeld, is de leeftijd.

Treinmiles worden verdeeld op volgorde van toenemende leeftijd, wat betekent dat eerst berekend wordt hoeveel treinmiles Piet krijgt. Hij is de enige in zijn leeftijdsgroep, dus de woonregio factor kan hier buiten beschouwing gelaten worden. Wat nu wel van belang is, is het maximaal aantal te ontvangen treinmiles. Als er geen maximum zou zijn, zou Piet namelijk alle 1800 beschikbare treinmiles ontvangen. Door het maximum ontvangt hij dus 500 treinmiles.

Nu Piet's treinmiles zijn uitgerekend, blijven er nog 1300 treinmiles over om te verdelen in de volgende leeftijdsgroep. Deze bestaat uit 2 mensen: Jan en Klaas. Nu speelt de woonregio factor wel een belangrijke rol. Samen hebben Jan en Klaas 3 woonregio factoren. Als we de overgebleven 1300 treinmiles op basis van de woonregio factor verdelen, betekent dit dat Klaas met woonregio factor 1 recht heeft op $1/3 * 1300 = 433$ treinmiles en Jan met woonregio factor 2 recht heeft op $2/3 * 1300 = 866$ treinmiles (let op, er wordt naar beneden afgerond). Het maximum aantal treinmiles dat beiden mogen ontvangen, is 500. Dat betekent dat Jan 500 treinmiles krijgt in plaats van 866. Het verschil van 366 treinmiles blijft in de te verdelen pot zitten.

Na het verdelen van treinmiles aan Jan en Klaas bevat de pot nog 367 treinmiles (1800-500-500-433), nu gaan er geen treinmiles verloren door de afronding naar beneden die we in de berekening hiervoor hebben gedaan). Dit wordt verdeeld over de volgende leeftijdsgroep die weer twee personen bevat: Nel en Ria. Dat betekent dat ook hier weer gerekend moet worden met de woonregio factor die hier opgeteld op 4 uitkomt. Nel heeft daarbij recht op $3/4 * 367 = 275$ treinmiles en Ria heeft recht op $1/4 * 367 = 91$ treinmiles. Het maximaal aantal te ontvangen treinmiles voor deze leeftijdsgroep is 250, wat betekent dat Nel 250 treinmiles krijgt

en Ria, zoals berekend, 91.

Alle personen die recht hebben op treinmiles, hebben deze gekregen. Niet alle treinmiles die beschikbaar zijn, zijn ook daadwerkelijk verdeeld. Het restant is 26 treinmiles, waar nu niets meer mee wordt gedaan.

9.7.6 Resultaten bij gebruik van lege of uitzonderlijke waarden

Onderstaande tabel bevat het overzicht van de resultaten van de verdeling als één van de attributen een lege of uitzonderlijke waarde heeft.

Objecttype	Attribuut	Waarde	Resultaat
Verdeler	Verdeelplafond	Leeg	Leeg. Verdeling wordt niet toegepast
Ontvanger	Naar rato	Leeg	Fout
	Maximum aanspraak	Leeg	Fout
		Kleiner dan 0	Fout
	Criterium bij gelijke volgorde	Leeg	Fout

Tabel 19 Bijzondere waarden bij verdeling

N.B. Als de situatie uit bovenstaande tabel bij één van de ontvangers optreedt dan zal dat al tot het genoemde resultaat leiden.

9.7.7 Syntax specificatie

Syntax verdeling specificatie:

```

<verdeling> ::= <attribuutvanonderwerp> "wordt verdeeld over" <attribuutvanonderwerp> ",
waarbij wordt verdeeld" (<verdelenzondergroepen> | <meervoudigcriterium>)

<verdelenzondergroepen> ::= "in gelijke delen" | ("naar rato van"
<attribuutmetlidwoord>)

<meervoudigcriterium> ::= ":" \n (<verdelenovertgroepen> | (<verdelenzondergroepen> ","))
[\n <maximumaanspraak>] [\n <verdeelafronding>] [\n <onverdeelderest>]

<verdelenovertgroepen> ::= "- op volgorde van" (afnemende | toenemende)
<attribuutmetlidwoord> \n <criteriumbijgelijkevolgorde> ", "

<criteriumbijgelijkevolgorde> ::= "- bij even groot criterium" ("in gelijke delen" |
("naar rato van" <attribuutmetlidwoord>)) ", "

<maximumaanspraak> ::= "- met een maximum van" <attribuutmetlidwoord> ", "

<verdeelafronding> ::= "- afgerond op" <geheelgetal> "decimalen naar beneden."

<onverdeelderest> ::= "Als onverdeelde rest blijft" <attribuutvanonderwerp> "over."

```

9.8 Dagsoortdefinitie

Een dagsoortdefinitie wordt gebruikt om te definiëren wanneer een dag van een bepaalde dagsoort is. Een voorbeeld hierbij kan zijn om te bepalen of een dag een Kerstdag is, of dat een dag een andere feestdag is. Voor meer informatie wordt verwezen naar paragraaf 3.12.

In een dagsoortdefinitie worden voorwaarden opgenomen om concreet te bepalen wanneer een datum van die dagsoort is. Zo is een datum een feestdag wanneer het een nieuwjaarsdag, paasdag, koningsdag, hemelvaartsdag, pinksterdag of kerstdag is. Of een datum is een kerstdag wanneer de datum de 25^{ste} of 26^{ste} dag van de 12^{de} maand is. Het onderwerp van een dagsoortdefinitie is altijd "de/een" (willekeurige) dag. Een voorbeeld van een dagsoortdefinitie

voor de dagsoort kerstdag kan dan ook als volgt opgesteld worden:

Regel Kerstdag
 geldig altijd
 Een dag is een **kerstdag**
 indien de dag aan alle volgende voorwaarden voldoet:

- de maand uit (de dag) is gelijk aan 12
- de dag voldoet aan ten minste één van de volgende voorwaarden:
 - de dag uit (de dag) is gelijk aan 25
 - de dag uit (de dag) is gelijk aan 26.

Syntax dagsoortdefinitie specificatie:

```
<dagsoortdefinitie> ::= "Een dag is een " <dagsoortnaam>
<dagsoortnaam> ::= <karakterreeks>
```

9.9 Recursie

Bij toepassing van een gelijkstelling, kenmerktoekenning, feitcreatie en objectcreatie is een cyclische specificatie, waarbij een eigenschap (rol, kenmerk of attribuut) van een instantie van een objecttype wordt afgeleid van dezelfde eigenschap van **dezelfde** instantie van het objecttype, foutief.

Wel zijn binnen specifieke regelgroepen vormen van recursie toegestaan. Recursie houdt in dat een eigenschap van een instantie van een objecttype wordt afgeleid van dezelfde eigenschap van een **andere** instantie van het objecttype.

Het onderscheid tussen een echt cyclische en een recursieve specificatie is echter niet betrouwbaar geautomatiseerd te bepalen. Daarom worden regels die cyclisch zijn in principe afgekeurd. Als er echter sprake is van recursie, dan kan de moeten de regels in een regelgroep worden gegroepeerd en die regelgroep moet als recursief worden gekenmerkt.

In die regelgroep worden de regels opgenomen die worden toegepast op instanties van objecttypes die in de regelgroep worden gecreëerd. Deze regels maken gebruik van resultaten van dezelfde regels die zijn toegepast op de voorgaande instantie van het objecttype.

De regels in de regelgroep worden herhaaldelijk uitgevoerd, maar eindeloze herhaling wordt voorkomen door de voorwaarden die worden gesteld aan de objectcreatieregels.

Dit betekent dat een recursieve regelgroep moet voldoen aan de volgende voorwaarden:

- De regelgroep bevat een objectcreatieregels (zie par. 9.3).
- De objectcreatieregels bevat een voorwaarde met betrekking tot een maximale of minimale waarde van de af te leiden waarde waarboven c.q. waaronder de regel niet meer wordt uitgevoerd.
- De objectcreatieregels bevat een voorwaarde met betrekking tot het maximale aantal iteraties waarboven de regel niet meer wordt uitgevoerd.

Een recursieve regelgroep kan bijvoorbeeld worden gebruikt voor het afleiden van een bruto bedrag als het netto bedrag en het percentage dat op het te berekenen bruto bedrag moet worden ingehouden, vaststaan.

10 Voorwaardendeel

In het voorwaardendeel van een regel staan één of meerdere voorwaarden gespecificeerd, waaraan voldaan moet worden om de actie in het resultaatdeel uit te voeren. Voorwaarden zijn condities en worden gespecificeerd met predicaten (zie hoofdstuk 7).

De actie in het resultaatdeel wordt pas uitgevoerd als aan de voorwaarden is voldaan, dit wordt ook wel het vuren van de regel genoemd³³. Een voorwaardendeel wordt gespecificeerd door *indien* op te nemen in een regel waarna de voorwaarde gespecificeerd moet worden. Bij een enkele conditie spreken we over een **elementaire voorwaarde**, bij een samengestelde conditie spreken we over een **samengestelde voorwaarde**. Het voorwaardendeel wordt tot slot afgesloten met een punt (.).

Er moet verder onderscheid gemaakt worden tussen voorwaarden over enkelvoudige en meervoudige onderwerpexpressies. Een **enkelvoudige voorwaarde** controleert slechts één enkele instantie terwijl een **meervoudige voorwaarde** meerdere instanties controleert. Bij een meervoudige voorwaarde dient ook de onderwerpexpressie meervoudig te zijn, en moet het voorafgegaan worden door een kwantificatie.

Een voorbeeld van een enkelvoudige voorwaarde is het nagaan of een *Vlucht* een *belaste reis* is. Dit is het geval als de bestemming ook per trein bereikbaar is, oftewel als het attribuut *bereikbaar per trein* van de instantie van *Vlucht* die de rol *reis* de waarde *waar* heeft. Dit is een enkelvoudige conditie aangezien slechts één instantie gecontroleerd moeten worden. We kunnen in dit geval schrijven:

indien *bereikbaar per trein* van de *Vlucht* gelijk is aan *waar*

Een voorbeeld van een meervoudige voorwaarde is het nagaan of een *Vlucht* geen minderjarige passagiers bevat. Dit is het geval als er geen *passagier* is waarvan de leeftijd kleiner is dan de volwassenleeftijd. Dit is een meervoudige voorwaarde: alle instanties van *Natuurlijk persoon* die de rol *passagier* invullen voor de instantie van *Vlucht* moeten gecontroleerd worden. In dit geval kunnen we schrijven:

indien alle *leeftijden* van alle *passagiers* van de *Vlucht* groter zijn dan *volwassenleeftijd*

10.1 Elementaire voorwaarde

Een elementaire voorwaarde is een enkele voorwaarde waaraan voldaan moet worden om de actie in het resultaatdeel van een regel uit te voeren, of het is één van de voorwaarden in een samengestelde voorwaarde. Een elementaire voorwaarde wordt gespecificeerd door de enkele schrijfwijze bij voorwaarden van het te gebruiken predicaat (zie Tabel 16 van paragraaf 8.1) te gebruiken. Deze schrijfwijze is een vragende vorm; het werkwoord staat achteraan.

Een voorbeeld is de toekenning van het kenmerk *minderjarig* van *Natuurlijk persoon*. Deze toekenning moet alleen plaatsvinden als de *leeftijd* van een *Natuurlijk persoon* kleiner is dan de leeftijd waarop men volwassen wordt. In RegelSpraak kan dat als volgt opgeschreven worden:

³³ Het predicaat *is gevraagd* (zie paragraaf **Fout! Verwijzingsbron niet gevonden.**) kan gebruikt worden, om te controleren of dit bij een bepaalde regel het geval is geweest.

Parameter *volwassenleeftijd* : Numeriek (niet-negatief geheel getal) met eenheid jr

Regel kenmerktoekenning persoon minderjarig

geldig altijd

Een *Natuurlijk persoon* is *minderjarig*
indien zijn *leeftijd* kleiner is dan de *volwassenleeftijd*.

Een ander voorbeeld is de toekenning van het kenmerk *belaste reis* voor het objecttype *Vlucht*.

Dit is geen beziel objecttype, we moeten de expressie bij het predicaat dus anders formuleren.

Regel belaste reis

geldig altijd

Een *Vlucht* is een *belaste reis*
indien *bereikbaar per trein* van de *Vlucht* gelijk is aan *waar*.

Syntax voorwaardendeel specificatie:

Opmerking: deze syntaxspecificatie maakt reeds gebruik van RegelSpraak-concepten die in navolgende paragrafen behandeld worden.

```

<voorwaardendeel> ::= (("indien" | ("gedurende de tijd dat")
  (<topelelementairevoorwaarde> | <topevelsamengesteldevoorwaarde>)) |
  <periodevergelijkingenkelvoudig>

<topelelementairevoorwaarde> ::= <topelevelvoorwaardevergelijking> |
  <consistentievoorwaarde>

<topelevelvoorwaardevergelijking> ::= <topelevelgetalvergelijking> |
  <topelevelobjectvergelijking> | <topeleveltekstvergelijking> | <topeleveldatumvergelijking>
  | <topelevelbooleanvergelijking> | <periodevergelijkingelementair>

<consistentievoorwaarde> ::= "regelversie" <karakterreeks> "is" ("gevuurd" |
  "inconsistent")

<periodevergelijkingenkelvoudig> ::= ("vanaf" <datumexpressie>) | ("van"
  <datumexpressie> "tot" <datumexpressie>) | ("van" <datumexpressie> "tot en met"
  <datumexpressie>) | ("tot" <datumexpressie>) | ("tot en met" <datumexpressie>)

<topelevelgetalvergelijking> ::= <topeveleenzijdigegetalvergelijking> |
  <topeleveltweezijdigegetalvergelijking>

<topeveleenzijdigegetalvergelijking> ::= <getalexpressie>
  <topeveleenzijdigegetalvergelijkingoperator>

<topeleveltweezijdigegetalvergelijking> ::= <getalexpressie>
  <topeleveltweezijdigegetalvergelijkingoperator> <getalexpressie>

<topeveleenzijdigegetalvergelijkingoperator> ::=
  <topeveleenzijdigegetalvergelijkingoperatorenkelvoud> |
  <topeveleenzijdigegetalvergelijkingoperatormeervoud>

<topeveleenzijdigegetalvergelijkingoperatorenkelvoud> ::= ("leeg is" | "gevuld is") |
  "aan de elfproef voldoet"

<topeveleenzijdigegetalvergelijkingoperatormeervoud> ::= ("leeg zijn" | "gevuld zijn")
  | "aan de elfproef voldoen"

<topeleveltweezijdigegetalvergelijkingoperator> ::=
  <topeleveltweezijdigegetalvergelijkingoperatorenkelvoud> |
  <topeleveltweezijdigegetalvergelijkingoperatormeervoud>

<topeleveltweezijdigegetalvergelijkingoperatorenkelvoud> ::= "gelijk is aan" | "ongelijk
  is aan" | "groter is dan" | "groter of gelijk is aan" | "kleiner of gelijk is aan" |
  "kleiner is dan"

<topeleveltweezijdigegetalvergelijkingoperatormeervoud> ::= "gelijk zijn aan" |
  "ongelijk zijn aan" | "groter zijn dan" | "groter of gelijk zijn aan" | "kleiner of
  gelijk zijn aan" | "kleiner zijn dan"

```


10.2 Samengestelde voorwaarde

Bij een samengestelde voorwaarde is sprake van meerdere condities waaraan voldaan moet worden. Elke conditie is een elementaire of samengestelde voorwaarde. Deze voorwaarden vormen een lijst met bullets. De lijst moet voorafgegaan worden door de woorden *er aan* <kwantificatie> *volgende voorwaarden wordt voldaan*., of als de condities slechts één attribuut controleren <expressie> *aan* <kwantificatie> *volgende voorwaarden voldoet*. De samengestelde schrijfwijze van de condities moet hier gebruikt worden. Deze schrijfwijze is een stellende vorm; het werkwoord staat vooraan. De onderstaande voorbeelden zullen dit verder verduidelijken.

Een voorbeeld van een samengestelde voorwaarde met de kwantificatie 'alle' is als volgt:

Parameter de **pensioenleeftijd** : Numeriek (geheel getal) met eenheid jr

Parameter de **duurzaamheidskorting minimale afstand** : Numeriek (geheel getal) met eenheid km

Regel Recht op Duurzaamheidskorting
geldig altijd

Een **passagier** heeft **recht op duurzaamheidskorting**
indien hij aan alle volgende voorwaarden voldoet:

- zijn **reis** is **duurzaam**
- de **afstand tot bestemming in kilometers** van zijn **reis** is groter of gelijk aan de **duurzaamheidskorting minimale afstand**.
- zijn **leeftijd** is groter of gelijk aan de **pensioenleeftijd**

We kunnen bovenstaande regel ook specificeren met een aantal kenmerken van *Natuurlijk persoon* in plaats van het attribuut *leeftijd* door gebruik te maken van een geneste voorwaarde waarbij ieder niveau een bullet meer krijgt dan het bovenliggende niveau:

Regel Recht op Duurzaamheidskorting
geldig altijd

Een **passagier** heeft **recht op duurzaamheidskorting**
indien hij aan alle volgende voorwaarden voldoet:

- zijn **reis** is **duurzaam**
- de **afstand tot bestemming in kilometers** van zijn **reis** is groter of gelijk aan de **duurzaamheidskorting minimale afstand**
- hij voldoet aan geen van de volgende voorwaarden:
 - hij is een **passagier jonger dan 18 jaar**
 - hij is een **passagier van 18 tot en met 24 jaar**
 - hij is een **passagier van 24 tot en met 64 jaar**.

Samengestelde voorwaarden kunnen ook zo geformuleerd worden zodat ze alleen voorwaarden stellen aan een enkel attribuut. Dit kan middels het volgende (taal)patroon: <expressie> *aan* <kwantificatie> *volgende voorwaarden voldoet*.. Het objecttype *Natuurlijk persoon* heeft een attribuut *burgerservicenummer*. In het volgende voorbeeld moet dit attribuut *burgerservicenummer* aan twee voorwaarden voldoen.

Parameter de **burgerservicenummer lengte** : Numeriek (geheel getal)

Regel Geldig burgerservicenummer
geldig altijd

Een **Natuurlijk persoon** heeft een **geldig burgerservicenummer**
indien zijn **burgerservicenummer** aan alle volgende voorwaarden voldoet:

- het **burgerservicenummer** is numeriek met exact **9** cijfers
- het **burgerservicenummer** voldoet aan de elfproef.

In het volgende voorbeeld wordt bepaald of een natuurlijk persoon een passagier is van 18 tot en met 24 jaar. Hiervoor moet hij aan een aantal voorwaarden voldoen. In het voorbeeld zie je dit terug in de tekst *hij aan* Het woord "hij" verwijst hier terug naar de natuurlijk persoon.

Regel Passagier van 18 tm 24 jaar
geldig altijd

Een **Natuurlijk persoon** is een **passagier van 18 tot en met 24 jaar**
indien hij aan alle volgende voorwaarden voldoet:

- zijn **leeftijd** is groter of gelijk aan de **volwassenleeftijd**
- zijn **leeftijd** is kleiner of gelijk aan **24 jr**
- hij is een **passagier**.

Syntax samengestelde voorwaarde specificatie:

```

<toplevelsamengesteldevoorwaarde> ::= (<objectexpressie> | <referentie> | <aggregatie> |
"er") "aan" <voorwaardekwantificatie> "volgende voorwaarde"["n"] ("voldoet" | "voldoen"
| "wordt voldaan") ":" <samengesteldevoorwaardeonderdeel>

<genestesamengesteldevoorwaarde> ::= (<objectexpressie> | <referentie> | <aggregatie> |
"er") ("voldoet" | "voldoen" | "wordt voldaan") "aan" <voorwaardekwantificatie>
"volgende voorwaarde"["n"] ":" <samengesteldevoorwaardeonderdeel>

<voorwaardekwantificatie> ::= "alle" | "geen van de" | ("ten minste" | "ten hoogste" |
"precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")

<samengesteldevoorwaardeonderdeel> ::= \n \t <genestevoorwaarde> (\n
<genestevoorwaarde>)+

<genestevoorwaarde> ::= ("•")+ (<elementairevoorwaarde> |
<genestesamengesteldevoorwaarde>)
" )

```

10.3 Tijdsafhankelijke voorwaarde

Aan het resultaatdeel van een regel kan een tijdsafhankelijke conditie (zie paragraaf 8.4) worden toegevoegd. Dit is alleen mogelijk als het resultaat van de regel tijdsafhankelijk is (het attribuut of kenmerk dat in de regel wordt bepaald is gespecificeerd met een tijdlijn).

Als een tijdsafhankelijke conditie wordt toegevoegd als voorwaarde, dan betekent dat dat het resultaatdeel alleen wordt bepaald voor de in de voorwaarde opgegeven periode.

De tijdlijn van het attribuut waarvoor de waarde wordt afgeleid moet knips bevatten op alle punten waarop de voorwaarde knips heeft. Als de voorwaarde knips op elke dag kan bevatten, dan kan de tijdlijn van het resultaat geen tijdlijn voor elke maand zijn, maar moet ook een tijdlijn voor elke dag zijn.

Voorbeeld.

De waarde van het attribuut *te betalen belasting* wordt afgeleid uit de optelling van de waarden van de attributen *belasting op basis van afstand* en *belasting op basis van reisduur*.

Alle drie deze attributen hebben het datatype numeriek €/mnd met een tijdlijn "voor elke dag". De regel ziet er als volgt uit:

De *te betalen belasting* van een *passagier* moet berekend worden als zijn *belasting op basis van afstand* plus zijn *belasting op basis van reisduur*

Als de invoer voor deze regel bestaat uit:

- Belasting op basis van afstand
 - €12/mnd van 1-1-2024 tot 1-1-2025
 - €18/mnd van 1-1-2025 tot 1-1-2026
- Belasting op basis van reisduur
 - €10/mnd van 1-1-2024 tot 1-1-2026

dan is het resultaat van deze regel:

- Te betalen belasting
 - €22/mnd van 1-1-2024 tot 1-1-2025
 - €28/mnd van 1-1-2025 tot 1-1-2026

Aan deze regel kan de specifieke tijdsafhankelijke conditie (zie paragraaf 8.4.3) worden toegevoegd dat het resultaat alleen moet worden afgeleid voor de periode van 1-1-2024 tot 8-2-2025. Die regel ziet er dan zo uit:

De **te betalen belasting** van een **passagier** moet berekend worden als zijn **belasting op basis van afstand** plus zijn **belasting op basis van reisduur** van dd. **1-1-2024** tot dd. **8-2-2025**

De voorwaarde staat niet na een “indien” in verband met de leesbaarheid van de regel, maar geldt wel als voorwaarde voor het af te leiden resultaat. Het resultaat van deze regel wordt:

- Te betalen belasting
 - €22/mnd van 1-1-2024 tot 1-1-2025
 - €28/mnd van 1-1-2025 tot 8-2-2025

Zoals in paragraaf 8.4.18.4 beschreven wordt in andere gevallen dan bij de enkelvoudige specifieke tijdsafhankelijke conditie (zoals in het voorbeeld hierboven) de “indien” vervangen door “gedurende de tijd dat”.

Bij gebruik hiervan ziet de regel er bijvoorbeeld zo uit:

Regel Belastingvermindering
geldig altijd

De **te betalen belasting** van een **passagier** moet berekend worden als zijn **belasting op basis van afstand** plus zijn **belasting op basis van reisduur** gedurende de tijd dat **hij** een **recht op belastingvermindering** heeft.

Nu is de periode waarin de passagier het kenmerk *recht op belastingvermindering* heeft, de periode waarvoor de regel een resultaat voor het attribuut *te betalen belasting* oplevert.

11 Variabelendeel

Na het voorwaardendeel kunnen **variabelen** gedefinieerd worden in het zogenaamde variabelendeel. Een variabele betreft de specificatie van een bepaalde waarde die alleen binnen de regel waarin de variabele is opgenomen geldt en wordt gebruikt. Door variabelen te gebruiken kunnen regels overzichtelijker en leesbaarder gemaakt worden en kunnen berekeningen waarvan de uitkomst eventueel op meerdere plekken in de regel wordt gebruikt op een centrale plek worden beschreven.

Een variabele wordt gedefinieerd met een expressie. Voor uitgebreidere informatie hierover wordt verwezen naar hoofdstuk 5, voor nu volstaat de opmerking dat een variabele een enkele attribuutwaarde kan zijn maar ook het resultaat van een berekening of logische bewerking.

Als er variabelen worden gebruikt in een regel, dan staan ze gespecificeerd achter de woorden *Daarbij geldt:* aan het eind van een regelversie. De specificatie van een variabele bestaat uit de naam van de variabele (eventueel voorafgegaan door een bepaald lidwoord), gevolgd door het woord *is* en tot slot de expressie die de variabele specificeert. Ter illustratie volgt hieronder een regel om de belasting op basis van afstand te bepalen, waarbij gebruik gemaakt wordt van twee variabelen X en Y:

Regel belasting op basis van afstand
geldig vanaf 2018

De *belasting op basis van afstand* van een *passagier* moet gesteld worden op X min Y indien hij aan alle volgende voorwaarden voldoet:

- zijn *reis* is een *belaste reis*
- hij voldoet aan ten minste één van de volgende voorwaarden:
 - hij is een *passagier jonger dan 18 jaar*
 - hij is een *passagier van 25 tot en met 64 jaar*
- de *afstand tot bestemming in kilometers* van zijn *reis* is groter dan 0
- de *afstand tot bestemming in kilometers* van zijn *reis* is kleiner of gelijk aan de *bovengrens afstand eerste schijf*
- X min Y is groter of gelijk aan 0

Daarbij geldt:

- X is het *lage basistarief eerste schijf*
- Y is het *lage tarief vermindering eerste schijf* maal de *afstand tot bestemming in kilometers* van zijn *reis*.

Syntax variabelendeel specificatie:

```
<variabelendeel> ::= "Daarbij geldt:" (\n \t <variabeleonderdeel>)* "."
<variabeleonderdeel> ::= [ <bepaaldlidwoord> ] <variabelenaam> "is" <expressie>
<variabelenaam> ::= <karakterreeks>
```

11.1 Rekenvolgorde variabelendeel

Bij het opstellen van RegelSpraak-regels zal rekening gehouden moeten worden met de volgorde waarin enerzijds voorwaarden uit het voorwaardendeel (zie hoofdstuk 109.9) worden gecontroleerd en anderzijds variabelen uit het variabelendeel (zie hierboven) worden uitgerekend. Deze volgorde wordt namelijk bepaald door de onderlinge afhankelijkheden (welke voorwaarden gebruiken welke variabelen) en door de volgorde waarin de variabelen in de *Daarbij geldt:* sectie staan. De volgende stelregels zijn hierbij van toepassing:

1. Een expressie in het variabelendeel mag alleen verwijzen naar andere variabelen die eerder in de betreffende regel gedefinieerd zijn.
2. De voorwaarden in een regel worden eerst berekend, daarna het resultaatdeel.
3. De voorwaarden worden ook van links naar rechts (van boven naar beneden) gecontroleerd.

Er wordt dus een volgorde bepaald van variabelen en voorwaarden, zodanig dat

- de variabelen in de gespecificeerde volgorde staan,
- de enkele voorwaarden in de gespecificeerde volgorde staan,
- een variabele berekend wordt voordat die de eerste keer wordt gebruikt, en
- variabelen niet onnodig worden berekend.

Aan de hand van een voorbeeld zal deze volgorde toegelicht en geïllustreerd worden. Stel eens voor dat we te maken hebben met het volgende deel van een RegelSpraak-regel:

Een **a** van een **persoon** moet berekend worden als B plus C plus D indien er aan alle volgende voorwaarden wordt voldaan:

- A is ongelijk aan 0
- B is groter dan 0.

Daarbij geldt:

A is zijn **a**
 B is zijn **b** gedeeld door A
 C is zijn **c** maal 100
 D is zijn **d**.

Hierbij is de volgende volgorde van toepassing conform bovenstaande beschrijving:

1. A is zijn **a**
2. A is ongelijk aan 0
3. B is zijn **b** gedeeld door A
4. B is groter dan 0
5. C is zijn **c** maal 100
6. D is zijn **d**
7. B plus C plus D

Hierbij worden

- de laatste vijf stappen alleen gedaan als A ongelijk is aan 0 (de voorwaarde van de tweede stap) waardoor een eventuele deling door 0 wordt voorkomen in de derde stap, en
- de laatste drie stappen alleen gedaan als B groter is dan 0 (de voorwaarde van de vierde stap).

12 Beslistabellen

In dit hoofdstuk wordt het gebruik van beslistabellen toegelicht waarin RegelSpraak wordt gebruikt. Een **beslistabel** is een weergave van één of meerdere RegelSpraakregels in één tabel. Deze regels moeten hetzelfde onderwerp en/of voorwaarden hebben om in dezelfde tabel te kunnen worden weergegeven. Doordat deze overeenkomstige delen van verschillende regels in een beslistabel maar één keer worden weergegeven, wordt de leesbaarheid van de regels vergroot. Andere voordelen zijn dat de samenhang van de regels duidelijk naar voren komt, en dat in een beslistabel de grammatica die RegelSpraak gebruikt ook kan worden gebruikt.

In een beslistabel kunnen alleen gelijkstellingen (zie paragraaf 9.1) en kenmerktoekenningen (zie paragraaf 9.2) worden weergegeven.

Het patroon voor de naamgevingsconstructie van een beslistabel is structureel hetzelfde als voor een RegelSpraak-regel (zie paragraaf 4.1). De enige afwijking hiervan is dat de naamgevingsconstructie van een RegelSpraak beslistabel altijd begint met het (vaste) woord *Beslistabel*, gevolgd door een zelf te kiezen naam.

Beslistabel woonregio factor

Het patroon voor de regelversie van een beslistabel is hetzelfde als voor een RegelSpraak-regel, zie hiervoor paragraaf 4.2.

Beslistabel woonregio factor
geldig altijd

Een voorbeeld van RegelSpraakregels met overeenkomstige delen zijn de regels omtrent de Woonregio factor:

Regel woonregio factor 1
geldig altijd

De **woonregio factor** van een **Natuurlijk persoon** moet gesteld worden op 1 indien er aan tenminste één van de volgende voorwaarden wordt voldaan:

- zijn **woonprovincie** is gelijk aan **Friesland**
- zijn **woonprovincie** is gelijk aan **Groningen**
- zijn **woonprovincie** is gelijk aan **Drenthe**
- zijn **woonprovincie** is gelijk aan **Zeeland**
- zijn **woonprovincie** is gelijk aan **Limburg**.

Regel woonregio factor 2
geldig altijd

De **woonregio factor** van een **Natuurlijk persoon** moet gesteld worden op 2 indien er aan tenminste één van de volgende voorwaarden wordt voldaan:

- zijn **woonprovincie** is gelijk aan **Noord-Brabant**
- zijn **woonprovincie** is gelijk aan **Gelderland**
- zijn **woonprovincie** is gelijk aan **Overijssel**
- zijn **woonprovincie** is gelijk aan **Flevoland**.

Regel woonregio factor 3
geldig altijd

De **woonregio factor** van een **Natuurlijk persoon** moet gesteld worden op 3 indien er aan tenminste één van de volgende voorwaarden wordt voldaan:

- zijn **woonprovincie** is gelijk aan **Noord-Holland**
- zijn **woonprovincie** is gelijk aan **Zuid-Holland**
- zijn **woonprovincie** is gelijk aan **Utrecht**.

Alle drie deze regels hebben een resultaatdeel met een gelijkstelling die de *woonregio factor van een Natuurlijk persoon* gelijkstelt aan een waarde en een voorwaardendeel dat condities stelt aan de *woonprovincie*. Omdat alleen de waardes van de gelijkstelling en de waardes van de voorwaarden van elkaar verschillen, kunnen deze RegelSpraakregels ook worden weergegeven in een beslistabel. Hierbij is de eerste kolom de nummering van de rijen in de tabel, de tweede

kolom het resultaatdeel en de derde kolom het voorwaardendeel. De kolom of kolommen waar het resultaatdeel in staat, wordt ook wel de **conclusie-kolom** genoemd. Vergelijkbaar wordt de kolom of de kolommen waar het voorwaardendeel in staat de **conditie-kolom** genoemd.

In algemene zin bestaat een beslistabel uit één kolom met nummering gevolgd door één of meerdere conclusie-kolommen en één of meerdere conditie-kolommen. De conclusie-kolom(men) kan zowel voor als na de conditie-kolom(men) geplaatst worden.

Beslistabel woonregio factor
geldig altijd

	de woonregio factor van een Natuurlijk persoon moet gesteld worden op	indien zijn woonprovincie gelijk is aan
1	1	'Friesland', 'Groningen', 'Drenthe', 'Zeeland' of 'Limburg'
2	2	'Noord-Brabant', 'Gelderland', 'Overijssel' of 'Flevoland'
3	3	'Noord-Holland', 'Zuid-Holland' of 'Utrecht'

Zoals in bovenstaand voorbeeld te zien is, staat in iedere conditie-kolom één voorwaarde die leidt tot de conclusie-kolom.

Een cel in de conditie-kolom kan een concatenatie-expressie (zie paragraaf 5.7) bevatten als het predicaat van de voorwaarde “is gelijk aan” of “is ongelijk aan” is. De tabel hierboven bevat daarvan een voorbeeld. De reeks met provincies in de conditie-kolom die dezelfde conclusie-kolom waarde krijgen, zijn in dezelfde cel gezet, gescheiden door een komma en voor de laatste provincie het woord “of”.

N.B. Hier wordt “of” gebruikt omdat bij het predicaat “is gelijk aan” er hooguit één waarde uit de concatenatie kan zijn waar aan wordt voldaan. In andere gevallen wordt bij een concatenatie het woord “en” gebruikt bij het formuleren van de reeks.

In een beslistabel bevat het resultaatdeel (de conclusie-kolom) van een gelijkstelling altijd het zinsdeel *moet gesteld worden op*, het is niet mogelijk om hier *wordt berekend als* te gebruiken, wat wel het geval is bij RegelSpraakregels. In de ene resultaatregel kan namelijk een vaste waarde toegekend worden, maar in een andere een berekende waarde. Daarom is ervoor gekozen om altijd *moet gesteld worden op* te gebruiken. Onderstaande beslistabel geeft hiervan een fictief voorbeeld.

Beslistabel woonregio factor
geldig altijd

	de woonregio factor van een Natuurlijk persoon moet gesteld worden op	indien zijn woonprovincie gelijk is aan
1	1	'Friesland'
2	1 + treinmiles op basis van evenredige verdeling	'Noord-Brabant'

De beslistabel voor Woonregio factor heeft 2 kolommen (de kolom voor de nummering niet mee tellende), maar het is mogelijk om meer kolommen te hebben. Dit is afhankelijk van het resultaatdeel en het voorwaardendeel van de RegelSpraakregels die worden weergegeven in de beslistabel. Wanneer in een beslistabel meerdere RegelSpraakregels worden genoteerd die

verschillende gelijkstellingen of kenmerktoekenningen hebben, maar wel dezelfde voorwaarden hanteren, dan wordt het aantal conclusie-kolommen groter dan 1. Wanneer in een beslistabel meerdere RegelSpraakregels worden genoteerd die dezelfde gelijkstelling of kenmerktoekenning hebben, maar meerdere voorwaarden kennen, dan wordt het aantal conditie-kolommen groter dan 1.

In de regels voor *Woonregio factor* wordt alleen gebruik gemaakt van de voorwaarde 'gelijk is aan', waardoor één conditie-kolom nodig is. In onderstaande regel voor *Belasting op basis van reisduur Tweede schijf* worden twee voorwaarden gebruikt: 'is groter dan' en 'is kleiner of gelijk aan'.

Regel Belasting op basis van reisduur Tweede schijf

geldig altijd

De *belasting op basis van reisduur* van een *passagier* moet berekend worden als het *percentage reisduur tweede schijf* van zijn *belasting op basis van afstand* naar beneden afgerond op 0 decimalen

indien hij aan alle volgende voorwaarden voldoet:

- de *reisduur per trein in minuten* van zijn *reis* is groter dan de *bovengrens reisduur eerste schijf*
- de *reisduur per trein in minuten* van zijn *reis* is kleiner of gelijk aan de *bovengrens tweede schijf*.

Naast deze regel voor *Belasting op basis van reisduur Tweede schijf*, kunnen we op vergelijkbare wijze regels voor *Belasting op basis van reisduur Eerste schijf* en *Belasting op basis van reisduur Derde schijf* opstellen. Deze zijn hier niet uitgeschreven, maar wel verwerkt in onderstaande beslistabel.

Omdat bovenstaande regel twee voorwaarden heeft, moeten deze in verschillende conditie-kolommen worden weergegeven. In de eerste kolom van de beslistabel wordt zoals gebruikelijk de nummering van het aantal rijen van de beslistabel weergegeven en in de tweede kolom het resultaatdeel. Het voorwaardendeel moet hier worden opgesplitst omdat we twee soorten voorwaarden hebben, die apart worden weergegeven in kolom 3 en 4. Voor iedere regelrij moet aan alle weergegeven voorwaarden voldaan worden. Bij regelrij 2 moet *de reisduur per trein in minuten* groter zijn dan *de bovengrens reisduur eerste schijf* én moet *de reisduur per trein in minuten* kleiner of gelijk zijn aan *de bovengrens reisduur tweede schijf*, zoals gesteld in de regel *Belasting op basis van reisduur Tweede schijf* die hierboven is uitgeschreven.

Beslistabel Belasting op basis van reisduur
geldig vanaf 2018

	de <i>belasting op basis van reisduur</i> van een <i>passagier</i> moet gesteld worden op	indien de <i>reisduur per trein in minuten</i> van zijn <i>reis</i> groter is dan	indien de <i>reisduur per trein in minuten</i> van zijn <i>reis</i> kleiner of gelijk is aan
1	het <i>percentage reisduur eerste schijf</i> van zijn <i>belasting op basis van afstand</i> naar beneden afgerond op 0 decimalen	n.v.t.	de <i>bovengrens reisduur eerste schijf</i>
2	het <i>percentage reisduur tweede schijf</i> van zijn <i>belasting op basis van afstand</i> naar beneden afgerond op 0 decimalen	de <i>bovengrens reisduur eerste schijf</i>	de <i>bovengrens reisduur tweede schijf</i>
3	het <i>percentage reisduur derde schijf</i> van zijn <i>belasting op basis van afstand</i> naar beneden afgerond op 0 decimalen	de <i>bovengrens reisduur tweede schijf</i>	n.v.t.

Bovenstaande voorbeelden bevatten allemaal conclusiekolommen op basis van een gelijkstelling. Het is ook mogelijk om een kenmerktoekenning op te nemen. Onderstaande tabel bevat een voorbeeld van de manier waarop dat in een beslistabel wordt vastgelegd.

	een passagier is een passagier jonger dan 18 jaar	indien zijn leeftijd kleiner is dan
1	waar	18 jr

Daarnaast is het ook mogelijk om in de conditiekolommen een voorwaarde met betrekking tot een kenmerk op te nemen. In onderstaande beslistabel is voorbeeld van een conditiekolom toegevoegd met een voorwaarde die betrekking heeft op een kenmerk.

	de belasting op basis van reisduur van een passagier moet gesteld worden op	indien hij een recht op duurzaamheidskorting heeft	indien de reisduur per trein in minuten van zijn reis groter is dan	indien de reisduur per trein in minuten van zijn reis kleiner of gelijk is aan
1	het percentage reisduur eerste schijf van zijn belasting op basis van afstand naar beneden afgerond op 0 decimalen	waar	n.v.t.	de bovengrens reisduur eerste schijf
2	het percentage reisduur tweede schijf van zijn belasting op basis van afstand naar beneden afgerond op 0 decimalen	waar	de bovengrens reisduur eerste schijf	de bovengrens reisduur tweede schijf
3	het percentage reisduur derde schijf van zijn belasting op basis van afstand naar beneden afgerond op 0 decimalen	n.v.t.	de bovengrens reisduur tweede schijf	n.v.t.

Het is mogelijk zo veel conclusie- en conditie-kolommen toe te voegen als nodig is. Hierbij is het belangrijk het doel van beslistabellen, het leesbaarder en overzichtelijker maken van regels, niet uit het oog te verliezen. Wanneer meerdere conditie-kolommen gebruikt worden geldt er een EN-relatie tussen deze voorwaarden. Dat houdt in dat alle voorwaarden in een regel moeten voldoen om de actie van het resultaatdeel te laten gelden.

Syntax beslistabellen specificatie:

Opmerking: beslistabellen geven een andere weergave van RegelSpraak-regels dan in de paragrafen over gelijkstellingen (paragraaf 7.1) en het voorwaardendeel (paragraaf 8) is besproken. In algemene zin gebruikt een beslistabel dus dezelfde bouwblokken als een tekstuele regel alleen zijn deze op een andere manier geordend. Aangezien de grammatica hetzelfde is en de syntaxspecificatie hiervan al is benoemd in de paragrafen hiervoor, wordt de syntax hier niet opnieuw weergegeven.

13 Formele specificatie van de RegelSpraak Syntax

In dit hoofdstuk wordt de formele specificatie van de RegelSpraak syntax en de notatiewijze die hiervoor wordt gebruikt vastgelegd. Ook wordt kort ingegaan op de overwegingen die zijn gemaakt bij de keuze van de notatiewijze.

Om een formele taal te beschrijven wordt een metataal gebruikt die de grammatica van de formele taal specificeert. De meest gebruikte notatiewijze voor zo'n metataal is de '**Backus-Naur Form**' (BNF). De originele BNF stamt uit 1959 maar ondertussen bestaan er veel varianten met ieder een eigen alfabet en uitbreidingen. Deze uitgebreidere varianten worden vaak 'Extended' BNF (EBNF) of 'Augmented' BNF (ABNF) genoemd maar er bestaat geen consensus over een standaard EBNF- of ABNF-notatie.

Ook organisaties die zich bezighouden met standaardisatie (W3C, ISO/IEC, OMG) maken geen eenduidig gebruik van hun eigen notatiewijze. Zo is de notatiewijze die voor XML gebruikt wordt³⁴ anders dan de EBNF notatiewijze zoals W3C deze voorlegt³⁵ en is de notatiewijze die gebruikt wordt voor het beschrijven van de Ada programmeertaal³⁶ ook anders dan de EBNF notatiewijze beschreven in ISO/IEC 14977³⁷. Een échte EBNF standaard bestaat daarom niet. De meeste organisaties leggen daarom taalspecificaties vast in hun eigen EBNF notatiewijze. Zie bijvoorbeeld hoofdstuk 2 uit de CORBA scripting taal zoals beschreven door de Object Management Group³⁸.

Om een idee te krijgen van de verschillen tussen bovengenoemde EBNF versies staan in Bijlage 2: de EBNF alfabetten van CORBA(OMG), W3C en ISO/IEC 14977 de alfabetten van drie versies tegen elkaar uitgezet en is in Bijlage 3: basisstructuur RegelSpraak in 3 EBNF varianten de basisstructuur van RegelSpraak als voorbeeld opgenomen om de drie EBNF varianten te vergelijken. De drie genoemde versies zijn allemaal voldoende expressief om de grammatica van RegelSpraak te vatten.

De ISO-specificatie krijgt echter de meeste kritiek: men heeft voornamelijk last van het vele aantal komma's dat gebruikt moet worden om delen van de grammatica te concateneren^{39,40}. Dit leidt voornamelijk tot onleesbare specificaties en hoge foutgevoeligheid.

13.1 Gebruikte notatie

Voor de syntaxspecificatie van RegelSpraak is gekozen om de notatiewijze die de OMG voor de CORBA scripting taal heeft gebruikt als basis te nemen, en deze waar nodig uit te breiden met de W3C opties. Reden hiervoor is dat de CORBA scripting taal van de OMG reeds een standaard is, en dat de W3C een prima aanvulling is als zijnde de organisatie die de webstandaarden voor het wereldwijde web ontwerpt.

³⁴ [Extensible Markup Language \(XML\) 1.0 \(Fifth Edition\) \(w3.org\)](http://www.w3.org/XML/1.0/)

³⁵ [BNF Notation for syntax \(w3.org\)](http://www.w3.org/BNF/)

³⁶ [Ada 2012 Language Reference Manual \(ada-auth.org\)](http://ada-auth.org/Ada2012LanguageReferenceManual/)

³⁷ <https://www.cl.cam.ac.uk/~mgk25/iso-14977.pdf>

³⁸ [script_book.bk \(omg.org\)](http://script_book.bk.omg.org/)

³⁹ [Don't Use ISO/IEC 14977 Extended Backus-Naur Form \(EBNF\) \(dwheeler.com\)](http://dwheeler.com/Don'tUseISOIEC14977ExtendedBackusNaurFormEBNF/)

⁴⁰ [bnf-was-here.pdf \(grammarware.net\)](http://grammarware.net/bnf-was-here.pdf)

Naast de tekstuele EBNF notatie wordt ook een grafische weergave van iedere EBNF regel opgenomen in de specificatie middels zogenaamde syntaxdiagrammen. Zie hiervoor verder het document “RegelSpraak specificatie - syntaxdiagrammen.docx”.

Er bestaan restricties in RegelSpraak die niet door een EBNF notatie opgevangen kunnen worden. In EBNF kun je bijvoorbeeld alleen aangeven dat er een referentie naar een objecttype moet staan maar niet dat de invulling van die referentie een verwijzing moet zijn naar een eerder genoemd objecttype. Zo kun je niet verwijzen naar een natuurlijk persoon terwijl het eerder in de regel over een vlucht ging. Ook kan bijvoorbeeld het aantal bullets dat gebruikt wordt bij geneste voorwaarden niet correct gespecificeerd worden. Dit aantal is namelijk afhankelijk van de nestingsdiepte, welke niet kan worden bijgehouden in de syntaxspecificatie.

Verder kunnen in RegelSpraak Unicode karakters gebruikt worden. Unicode omvat (in 2023) bijna 150 duizend karakters. Deze karakters zijn vanwege de omvang niet volledig uitgewerkt in de syntax specificatie.

In de rest van deze sectie wordt de voor deze specificatie gebruikte metataal notatiewijze in detail toegelicht. Hierbij wordt opgemerkt dat bij de syntaxdefinities spaties tussen de variabele delen (aangeduid met vishaken < en >) niet expliciet opgenomen zijn middels “ ”, ter bevordering van de leesbaarheid van deze syntaxdefinities. Uiteraard zullen deze wel toegepast moeten worden bij het opstellen van RegelSpraak-regels.

13.1.1 Terminale symbolen

Een terminaal symbool is een symbool dat onderdeel is van het alfabet van RegelSpraak. Dit kunnen enkele karakters zijn maar ook karakterreeksen. Terminale symbolen worden gebruikt om statische tekst in de beschreven syntax te specificeren. Ze worden in de gekozen notatiewijze omringd door aanhalingstekens.

Notatie:

`"Terminaal symbool"`

Voorbeelden:

- `"indien aan alle volgende voorwaarden wordt voldaan:"`
- `"+"`
- `"moet berekend worden als"`

13.1.2 Niet-terminale symbolen

Een **niet-terminaal symbool** is een symbool dat geen onderdeel van het alfabet van RegelSpraak is. Een niet-terminaal symbool kan plaatsvervangend gebruikt worden voor een reeks aan terminale en/of niet-terminale symbolen. Niet-terminale symbolen worden in de gekozen notatiewijze aangeduid met een karakterreeks omringt door kleiner- en groter-dan tekens, ook wel vishaken genoemd.

Notatie:

`<niet-terminaal symbool>`

Voorbeelden:

- `<voorwaarde>`
- `<objecttype>`
- `<gelijkstelling>`

13.1.3 Definitie

Een **definitie** beschrijft uit welke onderdelen een niet-terminaal symbool mag bestaan. Een definitie bestaat uit een niet-terminaal symbool, het “is gedefinieerd als” teken (`::=`) en een expressie. Deze expressie kan ook weer opgebouwd zijn uit andere terminale en niet-terminale symbolen, en uit andere symbolen uit de notatiewijze die hier beschreven wordt.

Notatie:

```
<niet-terminaal symbool> ::= <expressie>
```

Voorbeelden:

```
<regelversie> ::= <versie> <regelspraakregel>
```

13.1.4 Alternatief

Om te specificeren dat er in de syntax van RegelSpraak een keuze kan worden gemaakt tussen meerdere expressies wordt het **alternatief-symbool** (een verticale streep: '|') gebruikt. Dit symbool wordt tussen de verschillende alternatieven gezet. Exact één van deze opties moet gekozen worden.

Notatie:

```
|
```

Voorbeelden:

- <gelijkstelling> | <kenmerktoekenning> | <feitcreatie>
- "de" | "het" | "een"

13.1.5 Herhaling

Om **herhaling** van elementen aan te geven wordt een asterisk of plusteken aan het einde van het element gezet. Een asterisk geeft aan dat het voorgaande element 0, 1 of meer keer mag voorkomen en een plusteken geeft aan dat het voorgaande element 1 of meer keer moet voorkomen.

Notatie:

- ...*
- ...+

Voorbeelden:

- <voorwaarde>*
- <samengesteldevoorwaarde> ::= <type> <voorwaarde>+

13.1.6 Groepering

Om een **groepering** van elementen te specificeren worden haakjes gebruikt. Een groepering kan helpen bij het creëren van overzicht maar kan ook in combinatie met herhaling of alternativiteit gebruikt worden om herhaling of alternativiteit van een hele groep elementen aan te geven.

Notatie:

```
(...)
```

Voorbeelden:

- (<regelversie>)*
- ("voldoet" | "voldoen" | "wordt voldaan")

13.1.7 Optioneel

Blokhaken worden gebruikt om aan te geven dat een element, of groep aan elementen, **optioneel** is. Deze mogen dan nul of één keer voorkomen.

Notatie:

[...]

Voorbeelden:

`<regelversie> ::= <regelsoort> [voorwaarde]`

13.1.8 Controletekens

Controletekens worden gebruikt om witruimte in de syntax aan te geven, dit kan een horizontale tab zijn of een indicatie dat een nieuwe regel begint. De gebruikte controletekens zijn:

\n - nieuwe regel
 \t - horizontale tab
 \" - aanhalingsteken als escape karakter wanneer bedoeld als tekst en niet als openen of sluiten van terminaal symbool

Voorbeelden:

1. `<regel> ::= "Regel" <regelnaam> \n <regelversie> (\n <regelversie>)*`
2. `<regelversie> ::= <versie> \n <regelspraakregel>`

13.1.9 Overzicht gebruikte EBNF symbolen

De onderstaande tabel geeft een overzicht van de gebruikte EBNF symbolen.

Symbool	Functie
::=	Is gedefinieerd als
	Alternatief
"tekst"	Terminaal symbool d.w.z. enkele karakters of een karakterreeks welke onderdeel zijn van het RegelSpraak alfabet
<tekst>	Niet-terminaal symbool of Variabele d.w.z. geen onderdeel RegelSpraak alfabet
...*	Voorgaande element mag 0 of meer keer voorkomen
...+	Voorgaande element mag 1 of meer keer voorkomen
()	Groepering
[]	Optioneel - Ingesloten element mag 0 of 1 keer voorkomen
^(...)	Superscript – ingesloten deel wordt in superscript genoteerd.
\n	Nieuwe regel
\t	Horizontale tab
\"	Aanhalingsteken als escape karakter wanneer bedoeld als tekst en niet als terminaal symbool
\^	^ als escape karakter wanneer bedoeld als tekst en niet als superscript symbool

Tabel 20: de gebruikte EBNF symbolen.

13.2 Standaard syntax patronen

In deze sectie wordt een aantal generieke onderdelen van de RegelSpraak syntax vastgelegd. Deze onderdelen zullen bij veel RegelSpraak patronen terug gaan komen. Onderstaande nummering komt in combinatie met de paragraafnummering (namelijk 13.2) 1-op-1 terug in het al eerder genoemde separaat document met de bijbehorende syntaxdiagrammen.

```

1. <digit> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
2. <getal> ::= <geheelgetal> | <decimaalgetal> | <rationeelgetal>
3. <geheelgetal> ::= ["-"]<digit>+
4. <decimaalgetal> ::= ["-"]<digit>+ "," <digit>+
5. <rationeelgetal> ::= <geheelgetal>["_"]<geheelgetal>"/"<geheelgetal>
6. <letter> ::= "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l"
   | "m" | "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z" | "A"
   | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O"
   | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" | "á" | "à" | "â"
   | "ä" | "é" | "è" | "ê" | "ë" | "ö" | "ò" | "ó" | "ô" | "õ" | "ú" | "ù" | "ü" | "ï"
   | "i"
7. <leesteken> ::= " " | "." | "!" | "?" | ":" | ";" | "(" | ")" | "-"
<karakterreeks> ::= (<digit> | " " | <unicode>)+
8. <lidwoord> ::= <bepaaldlidwoord> | <onbepaaldlidwoord>
9. <bepaaldlidwoord> ::= "de" | "het"
10. <onbepaaldlidwoord> ::= "een"
11. <literalexpressie> ::= <booleanwaarde> | <getalwaarde> | <dedato> |
   <enumeratiewaarde> | <tekstwaarde>
12. <enumeratiewaarde> ::= "'" <karakterreeks> "'"
13. <tekstwaarde> ::= "\"" <karakterreeks> "\""
14. <booleanwaarde> ::= ("waar" | "onwaar")
15. <getalwaarde> ::= <getal> [(<eenheidsafkorting>)+ | (<eenheidsafkorting>+ "/"
   <eenheidsafkorting>)+]
16. <percentage> ::= <getal> "%"
17. <dedato> ::= "dd. "<datumwaarde> [<tijdwaarde>]
18. <datumwaarde> ::= <dag> "-" <maand> "-" <jaar>
19. <tijdwaarde> ::= <uur> ":" <minuut> ":" <seconde> "." <milliseconde>
20. <dag> ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "10" | "11" | "12"
   | "13" | "14" | "15" | "16" | "17" | "18" | "19" | "20" | "21" | "22" | "23" | "24"
   | "25" | "26" | "27" | "28" | "29" | "30" | "31"
21. <maand> ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "10" | "11" |
   "12"
22. <jaar> ::= ("0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9") ("0" | "1" |
   "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9") ("0" | "1" | "2" | "3" | "4" | "5" |
   "6" | "7" | "8" | "9") ("0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9")
23. <uur> ::= "00" | "01" | "02" | "03" | "04" | "05" | "06" | "07" | "08" | "09" |
   "10" | "11" | "12" | "13" | "14" | "15" | "16" | "17" | "18" | "19" | "20" | "21" |
   "22" | "23"
24. <minuut> ::= ("0" | "1" | "2" | "3" | "4" | "5") ("0" | "1" | "2" | "3" | "4" | "5"
   | "6" | "7" | "8" | "9")
25. <seconde> ::= ("0" | "1" | "2" | "3" | "4" | "5") ("0" | "1" | "2" | "3" | "4" |
   "5" | "6" | "7" | "8" | "9")
26. <milliseconde> ::= ("0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9") ("0"
   | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9") ("0" | "1" | "2" | "3" | "4"
   | "5" | "6" | "7" | "8" | "9")
27. <naamwoord> ::= [<bepaaldlidwoord>] <naam> ["(mv:" <meervoudsvorm> ")"]
28. <naam> ::= <karakterreeks>
29. <meervoudsvorm> ::= <karakterreeks>

```

13.3 Objecten en parameters

In deze sectie zal de syntax van het definiëren van objecttypen worden gespecificeerd. Ook hierbij geldt dat de toegepaste nummering in combinatie met de paragraafnummering (bijvoorbeeld 13.3.1) 1-op-1 terugkomt in het al eerder genoemde separaat document met de bijbehorende syntaxdiagrammen.

13.3.1 Objecttypen

```

1. <objecttypedefinitie> ::= "Objecttype" <naamwoord> ["(bezield)"] \n ((<koptekst> |
   <kenmerk> | <attribuut>) \n)+
2. <objecttypemetlidwoord> ::= <bepaaldlidwoord> <objecttypenaam>
3. <objecttypenaam> ::= <karakterreeks>

```

4. `<koptekst> ::= "---" <karakterreeks>`

13.3.2 Attributen en kenmerken

1. `<kenmerk> ::= (<naamwoord> "kenmerk" | <bezittelijkkenmerk> | <bijvoeglijkkenmerk>);"`
2. `<bezittelijkkenmerk> ::= <naamwoord> "kenmerk (bezittelijk)"`
3. `<bijvoeglijkkenmerk> ::= "is" <naam> ["(mv: " <meervoudsvorm> ")"] "kenmerk (bijvoeglijk)"`
4. `<kenmerknaam> ::= <karakterreeks>`
5. `<attribuut> ::= <naamwoord> \t (<datatype> | <domeinnaam>) ["gedimensioneerd met" <dimensienaam>];"`
6. `<attribuutmetlidwoord> ::= [<bepaaldlidwoord>] <attribuutnaam>`
7. `<attribuutnaam> ::= <karakterreeks>`

13.3.3 Datatypen

1. `<datatype> ::= <numeriekdatatype> | <percentagedatatype> | <tekstdatatype> | <booleandatatype> | <datumtijddatatype>`
2. `<numeriekdatatype> ::= "Numeriek (" <getalspecificatie> ")" ["met eenheid" [<eenheidmacht>+ | "1") "/" (<eenheidmacht>+)]`
3. `<percentagedatatype> ::= "Percentage (" <getalspecificatie> ")" ["met eenheid %" ["/" <eenheidsafkorting>]]`
4. `<tekstdatatype> ::= "Tekst"`
5. `<booleandatatype> ::= "Boolean"`
6. `<datumtijddatatype> ::= "Datum in dagen" | "Datum en tijd in millisecondes"`
7. `<getalspecificatie> ::= ["negatief" | "niet-negatief" | "positief"] ("geheel getal" | "getal met " <aantaldecimalen> " decimalen" | "getal")`
8. `<aantaldecimalen> ::= <positiefgeheelgetal>`

13.3.4 Domeinen

1. `<domeindefinitie> ::= "Domein" <domeinnaam> "is van het type" (<datatype> | <enumeratiespecificatie>)`
2. `<enumeratiespecificatie> ::= "Enumeratie" \n (\t <enumeratiewaarde> \n)+`
3. `<domeinnaam> ::= <karakterreeks>`

13.3.5 Eenheden

1. `<eenheidsysteem> ::= "Eenheidsysteem" <eenheidsysteemnaam> (\n <naamwoord> <eenheidsafkorting> [<omrekenspecificatie>])+`
2. `<omrekenspecificatie> ::= "=" ["1/"] <geheelgetal> <eenheidsafkorting>`
3. `<eenheidsysteemnaam> ::= <karakterreeks>`
4. `<eenheidsafkorting> ::= <karakterreeks>`
5. `<eenheidmacht> ::= <eenheidsafkorting> [^(<exponent>)]`
6. `<exponent> ::= <geheelgetal>`

13.3.6 Tijdlijnen

1. `<tijdlijn> ::= "voor" ("elke dag" | "elke maand" | "elk jaar")`

13.3.7 Dimensies

1. `<dimensie> ::= "Dimensie" <bepaaldlidwoord> <dimensienaam> ", bestaande uit de " <dimensienaammeervoud> <voorzetselspecificatie> \n (<labelwaardespecificatie> \n)+`
2. `<voorzetselspecificatie> ::= ("(na het attribuut met voorzetsel" ("van" | "in" | "voor" | "over" | "op" | "bij" | "uit") "):" | "(voor het attribuut zonder voorzetsel):")`
3. `<dimensienaam> ::= <karakterreeks>`
4. `<dimensienaammeervoud> ::= <karakterreeks>`
5. `<labelwaardespecificatie> ::= <digit>+ "." <dimensiewaarde>`
6. `<dimensiewaarde> ::= <karakterreeks>`

13.3.8 Parameters

1. `<parameterdefinitie> ::= "Parameter" <parametermetlidwoord> ":" (<datatype> | <domeinnaam>)`
2. `<parametermetlidwoord> ::= <bepaaldlidwoord> <parameternaam>`
3. `<parameternaam> ::= <karakterreeks>`

13.3.9 Feittypen

1. `<feittypedefinitie> ::= "Feittype" <feittypenaam> \n [<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n [<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n ("één" <rolnaam> | "meerdere" <meervoudrolnaam>) <relatiebeschrijving> ("één" <rolnaam> | "meerdere" <meervoudrolnaam>)`
2. `<wederkerigfeittypedefinitie> ::= "Wederkerig feittype" <feittypenaam> \n [<bepaaldlidwoord>] <rolnaam> ["(mv: " <meervoudrolnaam> ")"] \t <objecttypenaam> \n ("één" <rolnaam> <relatiebeschrijving> "één" <rolnaam>) | ("meerdere" <rolnaam> <relatiebeschrijving> "meerdere" <rolnaam>))`
3. `<feittypenaam> ::= <karakterreeks>`
4. `<rolnaam> ::= <karakterreeks>`
5. `<meervoudrolnaam> ::= <karakterreeks>`
6. `<relatiebeschrijving> ::= <karakterreeks>`

13.3.10 Dagsoort

1. `<dagsoort> ::= "Dagsoort" <naamwoord>`
2. `<dagsoortnaam> ::= <karakterreeks>`

13.4 RegelSpraak

13.4.1 Onderwerpketen

1. `<onderwerpketen> ::= ((<lidwoord> | "zijn") (<objecttypenaam> | <rolnaam> | <kenmerknaam>)) | ((<selector> "van" <onderwerpketen>) | <subselectie>)`

13.4.2 RegelSpraak-regel

1. `<regel> ::= "Regel" <regelnaam> (\n <regelversie>)+`
2. `<regelnaam> ::= <karakterreeks>`
3. `<regelversie> ::= <versie> \n <regelspraakregel>`
4. `<versie> ::= "geldig" <versiegeldigheid>`
5. `<versiegeldigheid> ::= "altijd" | ("vanaf " (<datumwaarde> | <jaar>) ["t/m " (<datumwaarde> | <jaar>)] | ("t/m " (<datumwaarde> | <jaar>))`
6. `<regelspraakregel> ::= <resultaatdeel> \n [<voorwaardendeel>] "." [<variabelendeel>]`
7. `<selector> ::= [<lidwoord>] <rolnaam>`
8. `<subselectie> ::= <onderwerpketen> ("die" | "dat") <predicaat>`
9. `<attribuutvanonderwerp> ::= [<kwantificatie>] <attribuutmetlidwoord> "van" <onderwerpketen>`
10. `<variabelendeel> ::= "Daarbij geldt:" (\n \t <variabeleonderdeel>)* "."`
11. `<variabeleonderdeel> ::= [<bepaaldlidwoord>] <variabelenaam> "is" <expressie>`
12. `<variabelenaam> ::= <karakterreeks>`

13.4.3 Resultaatdeel

1. `<resultaatdeel> ::= <gelijkstelling> | <kenmerktoekenning> | <objectcreatie> | <feitcreatie> | <consistentieregel> | <initialisatie> | <verdeling> | <dagsoortdefinitie>`

13.4.4 Gelijkstelling

1. `<gelijkstelling> ::= (<gelijkstellingtoekenning> | <gelijkstellingberekening>)`
2. `<gelijkstellingtoekenning> ::= <attribuutvanonderwerp> "moet gesteld worden op" <expressie>`
3. `<gelijkstellingberekening> ::= <attribuutvanonderwerp> "moet berekend worden als" (<getalexpressie> | <datumexpressie>)`

13.4.5 Kenmerktoekenning

1. `<kenmerktoekenning> ::= <onderwerpketen> ("is" | "heeft") ["een"] <kenmerknaam>`

13.4.6 ObjectCreatie

1. `<objectcreatie> ::= "Een" <onderwerpketen> "heeft een" <rolnaam> ["met" <waardetoekenning> [(", " <waardetoekenning>)* "en" <waardetoekenning>]]`
2. `<waardetoekenning> ::= <attribuutwaardetoekenning> | <kenmerkwaardetoekenning>`
3. `<attribuutwaardetoekenning> ::= <attribuut> "gelijk aan" <expressie>`
4. `<kenmerkwaardetoekenning> ::= <kenmerknaam> "gelijk aan" ("waar" | "onwaar")`

13.4.7 FeitCreatie

1. `<feitcreatie> ::= "Een" <rolnaam> "van een" <onderwerpketen> "is een" <rolnaam> "van een" <onderwerpketen>`

13.4.8 Consistentieregels

1. `<consistentieregel> ::= <enkelvoudigeconsistentieregel | <topevelsamengesteldcriterium> | <uniciteitscontrole>`
2. `<enkelvoudigeconsistentieregel> ::= <getalconsistentie> | <datumconsistentie> | <tekstconsistentie> | <objectconsistentie>`
3. `<getalconsistentie> ::= <getalexpressie> "moet" (<topeveleenzijdigegetalvergelijingsoperatormeervoud> | <topeveltweezijdigegetalvergelijingsoperatormeervoud>)`
4. `<datumconsistentie> ::= <datumexpressie> "moet" (<topeveleenzijdigedatumvergelijingsoperatormeervoud> | <topeveltweezijdigedatumvergelijingsoperatormeervoud>)`
5. `<tekstconsistentie> ::= <tekstexpressie> "moet" (<topeveleenzijdigetekstvergelijingsoperatormeervoud> | <topeveltweezijdigetekstvergelijingsoperatormeervoud>)`
6. `<objectconsistentie> ::= <objectexpressie> "moet" (<topeveleenzijdigeobjectvergelijingsoperatormeervoud> | <topeveltweezijdigeobjectvergelijingsoperatormeervoud>)`
7. `<topevelsamengesteldcriterium> ::= "er moet worden voldaan aan" ("het volgende criterium:" | (<consistentiekwantificatie> "volgende criteria:") <samengesteldcriteriumonderdeel>`
8. `<samengesteldcriteriumonderdeel> ::= \n \t <genestcriterium> (\n <genestcriterium>)+`
9. `<genestcriterium> ::= ("")+ (<voorwaardevergelijking> | <samengesteldcriterium>)`
10. `<samengesteldcriterium> ::= "er wordt voldaan aan" ("het volgende criterium:" | (<consistentiekwantificatie> "volgende criteria:") <samengesteldcriteriumonderdeel>`
11. `<uniciteitscontrole> ::= (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>) <vereniging>* "moeten uniek zijn."`
12. `<vereniging> ::= "verenigd met" (<alleattribuutvanonderwerp> | <uniciteitconcatenatie>)`
13. `<alleattribuutvanonderwerp> ::= "de" <meervoudsvorm> "van alle" ((<objecttypenaam> | <rolnaam>) "van" <onderwerpketen>) ["van" <onderwerpketen>]`
14. `<uniciteitconcatenatie> ::= "de concatenatie van" <meervoudsvorm> ("," <meervoudsvorm>)* "en" <meervoudsvorm> "van alle" ((<objecttypenaam> | <rolnaam>) "van" <onderwerpketen>) ["van" <onderwerpketen>]`

13.4.9 Initialisatie

1. `<initialisatie> ::= <attribuutvanonderwerp> "moet geïnitieerd worden op" <expressie>`

13.4.10 Verdeling

1. `<verdeling> ::= <attribuutvanonderwerp> "wordt verdeeld over" <attribuutvanonderwerp> ", waarbij wordt verdeeld" (<verdelenzondergroepen> | <meervoudigcriterium>)`
2. `<verdelenzondergroepen> ::= "in gelijke delen" | ("naar rato van" <attribuutmetlidwoord>)`
3. `<meervoudigcriterium> ::= ":" \n (<verdelenovergroepen> | (<verdelenzondergroepen> ",")) [\n <maximumaanspraak>] [\n <verdeelafronding>] [\n <onverdeelderest>]`
4. `<verdelenovergroepen> ::= "- op volgorde van" (afnemende | toenemende) <attribuutmetlidwoord> \n <criteriumbijgelijkevolgorde> ", "`
5. `<criteriumbijgelijkevolgorde> ::= "- bij even groot criterium" ("in gelijke delen" | ("naar rato van" <attribuutmetlidwoord>)) ", "`
6. `<maximumaanspraak> ::= "- met een maximum van" <attribuutmetlidwoord> ", "`
7. `<verdeelafronding> ::= "- afgerond op" <geheelgetal> "decimalen naar beneden."`
8. `<onverdeelderest> ::= "Als onverdeelde rest blijft" <attribuutvanonderwerp> "over."`

13.4.11 Dagsoortdefinitie

1. `<dagsoortdefinitie> ::= "Een dag is een" <dagsoortnaam>`

13.4.12 Voorwaardendeel

1. `<voorwaardendeel> ::= ("indien" (<topevelelementairevoorwaarde> | <topevelsamengesteldevoorwaarde>)) | <periodevergelijkingenkelvoudig>`
2. `<predicaat> ::= <elementairpredicaat> | <samengesteldpredicaat>`
3. `<elementairpredicaat> ::= <getalpredicaat> | <tekstpredicaat> | <datumpredicaat> | <objectpredicaat>`
4. `<samengesteldpredicaat> ::= "aan" <kwantificatie> "volgende voorwaarde"["n"]"`

- ```

("voldoet" | voldoen):" (<samengesteldevoorwaardeonderdeel> |
<toplevelvoorwaardevergelijking>)
5. <getalpredicaat> ::= <topleveltweezijdigegetalvergelijkingoperatormeervoud>
<getalexpressie>
6. <tekstpredicaat> ::= <topleveltweezijdigetekstvergelijkingoperatormeervoud>
<tekstexpressie>
7. <datumpredicaat> ::= <topleveltweezijdigedatumvergelijkingoperatormeervoud>
<datumexpressie>
8. <objectpredicaat> ::= <topleveltweezijdigeobjectvergelijkingoperatormeervoud>
<objectexpressie>

```

### 13.4.13 Samengestelde voorwaarde

- ```

1. <toplevelsamengesteldevoorwaarde> ::= (<objectexpressie> | <referentie> | <aggregatie>
| "er") "aan" <voorwaardekwantificatie> "volgende voorwaarde"["n"] ("voldoet" |
"voldoen" | "wordt voldaan") ":" <samengesteldevoorwaardeonderdeel>
2. <genestesamengesteldevoorwaarde> ::= (<objectexpressie> | <referentie> | <aggregatie>
| "er") ("voldoet" | "voldoen" | "wordt voldaan") "aan" <voorwaardekwantificatie>
"volgende voorwaarde"["n"] ":" <samengesteldevoorwaardeonderdeel>
3. <consistentiekwantificatie> ::= "alle" | "geen van de" | (("ten minste" | "ten
hoogste" | "precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")
4. <voorwaardekwantificatie> ::= "de" | "alle" | "geen van de" | (("ten minste" | "ten
hoogste" | "precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")
5. <kwantificatie> ::= "de" | "alle" | "al" | "geen van de" | (("ten minste" | "ten
hoogste" | "precies") (<getal> | "één" | "twee" | "drie" | "vier") "van de")
6. <samengesteldevoorwaardeonderdeel> ::= \n \t <genestevoorwaarde> (\n
<genestevoorwaarde>)+
7. <genestevoorwaarde> ::= ("•")+ (<elementairevoorwaarde> |
<genestesamengesteldevoorwaarde>)

```

13.4.14 Elementaire voorwaarde

- ```

1. <toplevelelementairevoorwaarde> ::= <toplevelvoorwaardevergelijking> |
<toplevelconsistentievoorwaarde>
2. <toplevelvoorwaardevergelijking> ::= <toplevelgetalvergelijking> |
<toplevelobjectvergelijking> | <topleveltekstvergelijking> |
<topleveldatumvergelijking> | <toplevelbooleaanvergelijking>
3. <toplevelgetalvergelijking> ::= <topleveleenzijdigegetalvergelijking> |
<topleveltweezijdigegetalvergelijking>
4. <topleveleenzijdigegetalvergelijking> ::= <getalexpressie>
<topleveleenzijdigegetalvergelijkingoperator>
5. <topleveltweezijdigegetalvergelijking> ::= <getalexpressie>
<topleveltweezijdigegetalvergelijkingoperator> <getalexpressie>
6. <topleveldatumvergelijking> ::= <topleveleenzijdigedatumvergelijking> |
<topleveltweezijdigedatumvergelijking>
7. <topleveleenzijdigedatumvergelijking> ::= <datumexpressie>
<topleveleenzijdigedatumvergelijkingoperator>
8. <topleveltweezijdigedatumvergelijking> ::= <datumexpressie>
<topleveltweezijdigedatumvergelijkingoperator> <datumexpressie>
9. <topleveltekstvergelijking> ::= <topleveleenzijdigetekstvergelijking> |
<topleveltweezijdigetekstvergelijking>
10. <topleveleenzijdigetekstvergelijking> ::= <tekstexpressie>
<topleveleenzijdigetekstvergelijkingoperator>
11. <topleveltweezijdigetekstvergelijking> ::= <tekstexpressie>
<topleveltweezijdigetekstvergelijkingoperator> <tekstexpressie>
12. <toplevelbooleaanvergelijking> ::= <topleveleenzijdigebooleaanvergelijking> |
<topleveltweezijdigebooleaanvergelijking>
13. <topleveleenzijdigebooleaanvergelijking> ::= <booleanexpressie>
<topleveleenzijdigebooleaanvergelijkingoperator>
14. <topleveltweezijdigebooleaanvergelijking> ::= <booleanexpressie>
<topleveltweezijdigebooleaanvergelijkingoperator> <booleanexpressie>
15. <toplevelobjectvergelijking> ::= <topleveleenzijdigeobjectvergelijking> |
<topleveltweezijdigeobjectvergelijking>
16. <topleveleenzijdigeobjectvergelijking> ::= <objectexpressie>
<topleveleenzijdigeobjectvergelijkingoperator>
17. <topleveltweezijdigeobjectvergelijking> ::= (<objectexpressie> | <referentie>)
<topleveltweezijdigeobjectvergelijkingoperator> <objectexpressie>
18. <toplevelconsistentievoorwaarde> ::= "regelversie" <karakterreeks> ("gevuurd" |
"inconsistent") "is"
19. <topleveleenzijdigegetalvergelijkingoperator> ::=
<topleveleenzijdigegetalvergelijkingoperatorenkelvoud> |
<topleveleenzijdigegetalvergelijkingoperatormeervoud>
20. <topleveleenzijdigegetalvergelijkingoperatorenkelvoud> ::=
[<geheleperiodevergelijking>] ("leeg is" | "gevuld is" | "aan de elfproef voldoet")

```

21. <topeleleenzijdigegetalvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("leeg zijn" | "gevuld zijn" | "aan de elfproef voldoen")
22. <topeleveltweezijdigegetalvergelijkingsoperator> ::=  
<topeleveltweezijdigegetalvergelijkingsoperatorenkelvoud> |  
<topeleveltweezijdigegetalvergelijkingsoperatormeervoud>
23. <topeleveltweezijdigegetalvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("gelijk is aan" | "ongelijk is aan" | "groter is dan" |  
"groter of gelijk is aan" | "kleiner of gelijk is aan" | "kleiner is dan")
24. <topeleveltweezijdigegetalvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("gelijk zijn aan" | "ongelijk zijn aan" | "groter zijn  
dan" | "groter of gelijk zijn aan" | "kleiner of gelijk zijn aan" | "kleiner zijn  
dan")
25. <topeleleenzijdigedatumvergelijkingsoperator> ::=  
<topeleleenzijdigedatumvergelijkingsoperatorenkelvoud> |  
<topeleleenzijdigedatumvergelijkingsoperatormeervoud>
26. <topeleleenzijdigedatumvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("leeg is" | "gevuld is" | ("een" <dagsoort> "is"))
27. <topeleleenzijdigedatumvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("leeg zijn" | "gevuld zijn" | ("een" <dagsoort>  
"zijn"))
28. <topeleveltweezijdigedatumvergelijkingsoperator> ::=  
<topeleveltweezijdigedatumvergelijkingsoperatorenkelvoud> |  
<topeleveltweezijdigedatumvergelijkingsoperatormeervoud>
29. <topeleveltweezijdigedatumvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("gelijk is aan" | "ongelijk is aan" | "later is dan" |  
"later of gelijk is aan" | "eerder of gelijk is aan" | "eerder is dan")
30. <topeleveltweezijdigedatumvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("gelijk zijn aan" | "ongelijk zijn aan" | "later zijn  
dan" | "later of gelijk zijn aan" | "eerder of gelijk zijn aan" | "eerder zijn dan")
31. <topeleleenzijdigetekstvergelijkingsoperator> ::=  
<topeleleenzijdigetekstvergelijkingsoperatorenkelvoud> |  
<topeleleenzijdigetekstvergelijkingsoperatormeervoud>
32. <topeleleenzijdigetekstvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("leeg is" | "gevuld is" | ("numeriek is met exact"  
"geheelgetal" "cijfers") | "aan de elfproef voldoet")
33. <topeleleenzijdigetekstvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("leeg zijn" | "gevuld zijn" | ("numeriek zijn met  
exact" <geheelgetal> "cijfers") | "aan de elfproef voldoen")
34. <topeleveltweezijdigetekstvergelijkingsoperator> ::=  
<topeleveltweezijdigetekstvergelijkingsoperatorenkelvoud> |  
<topeleveltweezijdigetekstvergelijkingsoperatormeervoud>
35. <topeleveltweezijdigetekstvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("gelijk is aan" | "ongelijk is aan")
36. <topeleveltweezijdigetekstvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("gelijk zijn aan" | "ongelijk zijn aan")
37. <topeleleenzijdigebooleanvergelijkingsoperator> ::=  
<topeleleenzijdigebooleanvergelijkingsoperatorenkelvoud> |  
<topeleleenzijdigebooleanvergelijkingsoperatormeervoud>
38. <topeleleenzijdigebooleanvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("leeg is" | "gevuld is")
39. <topeleleenzijdigebooleanvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("leeg zijn" | "gevuld zijn")
40. <topeleveltweezijdigebooleanvergelijkingsoperator> ::=  
<topeleveltweezijdigebooleanvergelijkingsoperatorenkelvoud> |  
<topeleveltweezijdigebooleanvergelijkingsoperatormeervoud>
41. <topeleveltweezijdigebooleanvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("gelijk is aan" | "ongelijk is aan")
42. <topeleveltweezijdigebooleanvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("gelijk zijn aan" | "ongelijk zijn aan")
43. <topeleleenzijdigeobjectvergelijkingsoperator> ::=  
<topeleleenzijdigeobjectvergelijkingsoperatorenkelvoud> |  
<topeleleenzijdigeobjectvergelijkingsoperatormeervoud>
44. <topeleleenzijdigeobjectvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ["een"] (<rolnaam> (<is" | "heeft")
45. <topeleleenzijdigeobjectvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ["een"] (<rolnaam> | <kenmerknaam>) ("zijn" | "hebben")
46. <topeleveltweezijdigeobjectvergelijkingsoperator> ::=  
<topeleveltweezijdigeobjectvergelijkingsoperatorenkelvoud> |  
<topeleveltweezijdigeobjectvergelijkingsoperatormeervoud>
47. <topeleveltweezijdigeobjectvergelijkingsoperatorenkelvoud> ::=  
[<geheleperiodevergelijking>] ("gelijk is aan" | "ongelijk is aan")
48. <topeleveltweezijdigeobjectvergelijkingsoperatormeervoud> ::=  
[<geheleperiodevergelijking>] ("gelijk zijn aan" | "ongelijk zijn aan")
49. <elementairevoorwaarde> ::= <voorwaardevergelijking> | <consistentievoorwaarde>

```

50. <voorwaardevergelijking> ::= <getalvergelijking> | <objectvergelijking> |
 <tekstvergelijking> | <datumvergelijking> | <booleanvergelijking> |
 <periodevergelijkingelementair>
51. <getalvergelijking> ::= <eenzijdigegetalvergelijking> | <tweezijdigegetalvergelijking>
52. <eenzijdigegetalvergelijking> ::= <getalexpressie>
 <eenzijdigegetalvergelijkingsoperator>
53. <tweezijdigegetalvergelijking> ::= <getalexpressie>
 <tweezijdigegetalvergelijkingsoperator> <getalexpressie>
54. <datumvergelijking> ::= <eenzijdigedatumvergelijking> | <tweezijdigedatumvergelijking>
55. <eenzijdigedatumvergelijking> ::= <datumexpressie>
 <eenzijdigedatumvergelijkingsoperator>
56. <tweezijdigedatumvergelijking> ::= <datumexpressie>
 <tweezijdigedatumvergelijkingsoperator> <datumexpressie>
57. <tekstvergelijking> ::= <eenzijdigetekstvergelijking> | <tweezijdigetekstvergelijking>
58. <eenzijdigetekstvergelijking> ::= <tekstexpressie>
 <eenzijdigetekstvergelijkingsoperator>
59. <tweezijdigetekstvergelijking> ::= <tekstexpressie>
 <tweezijdigetekstvergelijkingsoperator> <tekstexpressie>
60. <booleanvergelijking> ::= <eenzijdigebooleanvergelijking> |
 <tweezijdigebooleanvergelijking>
61. <eenzijdigebooleanvergelijking> ::= <booleanexpressie>
 <eenzijdigebooleanvergelijkingsoperator>
62. <tweezijdigebooleanvergelijking> ::= <booleanexpressie>
 <tweezijdigebooleanvergelijkingsoperator> <booleanexpressie>
63. <objectvergelijking> ::= <eenzijdigeobjectvergelijking> |
 <tweezijdigeobjectvergelijking>
64. <eenzijdigeobjectvergelijking> ::= <objectexpressie>
 <eenzijdigeobjectvergelijkingsoperator>
65. <tweezijdigeobjectvergelijking> ::= (<objectexpressie> | <referentie>)
 <tweezijdigeobjectvergelijkingsoperator> <objectexpressie>
66. <consistentievoorwaarde> ::= "regelversie" <karakterreeks> "is" ("gevuurd" |
 "inconsistent")
67. <periodevergelijking> ::= <periodevergelijkingenkelvoudig> |
 <periodevergelijkingelementair>
68. <periodevergelijkingenkelvoudig> ::= ("vanaf" <datumexpressie>) | ("van"
 <datumexpressie> "tot" <datumexpressie>) | ("van" <datumexpressie> "tot en met"
 <datumexpressie>) | ("tot" <datumexpressie>) | ("tot en met" <datumexpressie>)
69. <periodevergelijkingelementair> ::= "het is de periode"
 <periodevergelijkingenkelvoudig>
70. <geheleperiodevergelijking> ::= ["niet"] "gedurende" ("het gehele jaar" | "de gehele
 maand")
71. <eenzijdigegetalvergelijkingsoperator> ::=
 <eenzijdigegetalvergelijkingsoperatorenkelvoud> |
 <eenzijdigegetalvergelijkingsoperatormeervoud>
72. <eenzijdigegetalvergelijkingsoperatorenkelvoud> ::= ("is"
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")) | ("voldoet"
 [<geheleperiodevergelijking>] "aan de elfproef")
73. <eenzijdigegetalvergelijkingsoperatormeervoud> ::= ("zijn"
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")) | ("voldoen"
 [<geheleperiodevergelijking>] "aan de elfproef")
74. <tweezijdigegetalvergelijkingsoperator> ::=
 <tweezijdigegetalvergelijkingsoperatorenkelvoud> |
 <tweezijdigegetalvergelijkingsoperatormeervoud>
75. <tweezijdigegetalvergelijkingsoperatorenkelvoud> ::= "is"
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan" | "groter dan" | "groter
 of gelijk aan" | "kleiner of gelijk aan" | "kleiner dan")
76. <tweezijdigegetalvergelijkingsoperatormeervoud> ::= "zijn"
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan" | "groter dan" | "groter
 of gelijk aan" | "kleiner of gelijk aan" | "kleiner dan")
77. <eenzijdigedatumvergelijkingsoperator> ::=
 <eenzijdigedatumvergelijkingsoperatorenkelvoud> |
 <eenzijdigedatumvergelijkingsoperatormeervoud>
78. <eenzijdigedatumvergelijkingsoperatorenkelvoud> ::= ("is"
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")) | ([<geheleperiodevergelijking>]
 "een" <dagsoort> "is")
79. <eenzijdigedatumvergelijkingsoperatormeervoud> ::= ("zijn"
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")) | ([<geheleperiodevergelijking>]
 "een" <dagsoort> "zijn")
80. <tweezijdigedatumvergelijkingsoperator> ::=
 <tweezijdigedatumvergelijkingsoperatorenkelvoud> |
 <tweezijdigedatumvergelijkingsoperatormeervoud>
81. <tweezijdigedatumvergelijkingsoperatorenkelvoud> ::= "is"
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan" | "later dan" | "later of
 gelijk aan" | "eerder of gelijk aan" | "eerder dan")
82. <tweezijdigedatumvergelijkingsoperatormeervoud> ::= "zijn"

```

- [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan" | "later dan" | "later of gelijk aan" | "eerder of gelijk aan" | "eerder dan")
83. <eenzijdigetekstvergelijkingoperator> ::=  
 <eenzijdigetekstvergelijkingoperatorenkelvoud> |  
 <eenzijdigetekstvergelijkingoperatormeervoud>
84. <eenzijdigetekstvergelijkingoperatorenkelvoud> ::= ("is"  
 [<geheleperiodevergelijking>] ("leeg" | "gevuld" | ("numeriek met exact" <geheelgetal>  
 "cijfers")) | ("voldoet" [<geheleperiodevergelijking>] "aan de elfproef")
85. <eenzijdigetekstvergelijkingoperatormeervoud> ::= ("zijn"  
 [<geheleperiodevergelijking>] ("leeg" | "gevuld" | ("numeriek met exact" <geheelgetal>  
 "cijfers")) | ("voldoen" [<geheleperiodevergelijking>] "aan de elfproef")
86. <tweezijdigetekstvergelijkingoperator> ::=  
 <tweezijdigetekstvergelijkingoperatorenkelvoud> |  
 <tweezijdigetekstvergelijkingoperatormeervoud>
87. <tweezijdigetekstvergelijkingoperatorenkelvoud> ::= "is"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")
88. <tweezijdigetekstvergelijkingoperatormeervoud> ::= "zijn"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")
89. <eenzijdigebooleanvergelijkingoperator> ::=  
 <eenzijdigebooleanvergelijkingoperatorenkelvoud> |  
 <eenzijdigebooleanvergelijkingoperatormeervoud>
90. <eenzijdigebooleanvergelijkingoperatorenkelvoud> ::= "is"  
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")
91. <eenzijdigebooleanvergelijkingoperatormeervoud> ::= "zijn"  
 [<geheleperiodevergelijking>] ("leeg" | "gevuld")
92. <tweezijdigebooleanvergelijkingoperator> ::=  
 <tweezijdigebooleanvergelijkingoperatorenkelvoud> |  
 <tweezijdigebooleanvergelijkingoperatormeervoud>
93. <tweezijdigebooleanvergelijkingoperatorenkelvoud> ::= "is"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")
94. <tweezijdigebooleanvergelijkingoperatormeervoud> ::= "zijn"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")
95. <eenzijdigeobjectvergelijkingoperator> ::=  
 <eenzijdigeobjectvergelijkingoperatorenkelvoud> |  
 <eenzijdigeobjectvergelijkingoperatormeervoud>
96. <eenzijdigeobjectvergelijkingoperatorenkelvoud> ::= ("is" | "heeft")  
 [<geheleperiodevergelijking>] ["een"] <kenmerknaam>
97. <eenzijdigeobjectvergelijkingoperatormeervoud> ::= ("zijn" | "hebben")  
 [<geheleperiodevergelijking>] ["een"] <kenmerknaam>
98. <tweezijdigeobjectvergelijkingoperator> ::=  
 <tweezijdigeobjectvergelijkingoperatorenkelvoud> |  
 <tweezijdigeobjectvergelijkingoperatormeervoud>
99. <tweezijdigeobjectvergelijkingoperatorenkelvoud> ::= "is"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")
100. <tweezijdigeobjectvergelijkingoperatormeervoud> ::= "zijn"  
 [<geheleperiodevergelijking>] ("gelijk aan" | "ongelijk aan")

### 13.4.15 Berekening

1. <berekening> ::= <getalexpressie> ("plus" | "min" | "verminderd met" | "maal" |  
 "gedeeld door" | "gedeeld door (ABS)") <getalexpressie>

### 13.4.16 Expressie

1. <expressie> ::= <getalexpressie> | <objectexpressie> | <datumexpressie> |  
 <tekstexpressie> | <booleanexpressie> | <expressietussenhaakjes> |  
 <parametermetlidwoord> | <variabelenaam> | <concatenatie> | ("'" <enumeratiewaarde>  
 "'")
2. <concatenatie> ::= <expressie> [ ("", " <expressie>)\* ] (" en " | " of ") <expressie>
3. <getalexpressie> ::= <begrenzingexpressie> | <afrondingexpressie> | <getalfunctie> |  
 <getalaggregatie> | <attribuutvanonderwerp> | <parametermetlidwoord> | <variabelenaam>  
 | <getalwaarde> | <rekenjaar> | <jaaruitfunctie> | <maanduitfunctie> | <daguitfunctie>
4. <datumexpressie> ::= <datumfunctie> | <attribuutvanonderwerp> | <parametermetlidwoord>  
 | <variabelenaam> | <dedato> | <datumaggregatie>
5. <objectexpressie> ::= [ <kwantificatie> ] <onderwerpketen>
6. <tekstexpressie> ::= <tekstenwaardereeks> | <tekstwaarde> | <attribuutvanonderwerp> |  
 <parametermetlidwoord> | <variabelenaam>
7. <booleanexpressie> ::= <booleanwaarde> | <attribuutvanonderwerp> |  
 <parametermetlidwoord> | <variabelenaam>
8. <expressietussenhaakjes> ::= "(" <expressie> ")"
9. <tekstenwaardereeks> ::= "\" ("<expressie> ">") | <karakterreeks>)+ "\"
10. <functie> ::= <datumfunctie> | <getalfunctie>
11. <getalfunctie> ::= <percentagefunctie> | <wortelfunctie> | <minimalewaardefunctie> |  
 <maximalewaardefunctie> | <verminderdmetfunctie> | <tijdsduurtussen>

12. <percentagefunctie> ::= <getalexpressie> ["%"] "van" <getalexpressie>
13. <wortelfunctie> ::= "de wortel van" <getalexpressie> <afronding>
14. <machtsverheffenfunctie> ::= <getalexpressie> "tot de macht" <getalexpressie> <afronding>
15. <minimalewaardefunctie> ::= "de minimale waarde van" <getalexpressie> (" , " <getalexpressie>)\* "en " <getalexpressie>
16. <maximalewaardefunctie> ::= "de maximale waarde van" <getalexpressie> (" , " <getalexpressie>)\* "en " <getalexpressie>
17. <absolutewaardefunctie> ::= "de absolute waarde van (" <getalexpressie> ")"
18. <jaaruitfunctie> ::= "het jaar uit" <datumexpressie>
19. <maanduitfunctie> ::= "de maand uit" <datumexpressie>
20. <daguitfunctie> ::= "de dag uit" <datumexpressie>
21. <afrondingexpressie> ::= <getalexpressie> <afronding>
22. <afronding> ::= ("naar beneden" | "naar boven" | "rekenkundig" | "richting nul" | "weg van nul") "afgerond op" <geheelgetal> "decimalen"
23. <begrenzingexpressie> ::= <getalexpressie> " , " <begrenzing>
24. <begrenzing> ::= (<begrenzingminimum> | <begrenzingmaximum> | <begrenzingminimum> "en" <begrenzingmaximum>)
25. <begrenzingminimum> ::= "met een minimum van" <getalexpressie>
26. <begrenzingmaximum> ::= "met een maximum van" <getalexpressie>
27. <rekenjaar> ::= "Rekenjaar"
28. <tijdsduurtussen> ::= ("de tijdsduur van " | "de absolute tijdsduur van ") <datumexpressie> "tot" <datumexpressie> "in" ["hele"] <eenheidmeervoud>
29. <datumfunctie> ::= <datummet> | <eerstepaasdagvan> | <dedato> | <datumberekening> | <eerstevan> | <laatstevan> | <rekendatum>
30. <rekendatum> ::= "Rekendatum"
31. <datummet> ::= "de datum met jaar, maand en dag(" <getalexpressie> " , " <getalexpressie> " , " <getalexpressie> ")"
32. <eerstepaasdagvan> ::= "de eerste paasdag van (" <jaar> ")"
33. <datumberekening> ::= <datumexpressie> ("plus" | "min") <getalexpressie> <eenheidsafkorting>
34. <eerstevan> ::= "de eerste van" <datumexpressie> (" , " <datumexpressie>)\* "en" <datumexpressie>
35. <laatstevan> ::= "de laatste van" <datumexpressie> (" , " <datumexpressie>)\* "en" <datumexpressie>
36. <referentie> ::= <bezieldereferentie> | <nietbezieldereferentie> | <dagsoortreferentie>
37. <bezieldereferentie> ::= "hij" | ("zijn" <attribuutnaam>) | ("zijn" <rolnaam>)
38. <nietbezieldereferentie> ::= <objecttypemetlidwoord>
39. <dagsoortreferentie> ::= "de dag"
40. <aggregatie> ::= <getalaggregatie> | <datumaggregatie> | <dimensieaggregatie>
41. <getalaggregatie> ::= <getalaggregatiefunctie> (<expressie>)
42. <getalaggregatiefunctie> ::= "het aantal" | "de maximale waarde van" | "de minimale waarde van" | "de som van"
43. <datumaggregatie> ::= <datumaggregatiefunctie> (<expressie>)
44. <datumaggregatiefunctie> ::= "de eerste van" | "de laatste van"
45. <dimensieaggregatie> ::= (<getalaggregatiefunctie> | <datumaggregatiefunctie>) <attribuutvanonderwerp> <dimensieselectie>
46. <dimensieselectie> ::= "over" (<aggregerenoveralldimensies> | <aggregerenoververzameling> | <aggregerenoverbereik>) " , "
47. <aggregerenoveralldimensies> ::= "alle" <dimensienaammeervoud>
48. <aggregerenoververzameling> ::= "de" <dimensienaammeervoud> "vanaf" <dimensiewaarde> "t/m" <dimensiewaarde>
49. <aggregerenoverbereik> ::= "de" <dimensienaammeervoud> "in {" <dimensiewaarde> [{" , " <dimensiewaarde>}\* "en" <dimensiewaarde>] "}"
50. <conditiebijexpressie> ::= ("gedurende de tijd dat" (<toplevelelementairevoorwaarde> | <toplevelsamengesteldevoorwaarde>)) | <periodevergelijkingenkelvoudig>
51. <waardepertijdseenheidaggregatie> ::= "het totaal van" <expressie> [<conditiebijexpressie>]
52. <tellingaantaldagen> ::= "het aantal dagen in" ("de maand" | "het jaar") "dat" <expressie>
53. <tijdsevenredigdeel> ::= "het tijdsevenredig deel per" ("maand" | "jaar") "van" <expressie> <conditiebijexpressie>



## 14 Lijst met figuren

|                                                                                                                                                                                                                                                                                                                                                                |     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Figuur 1: een RegelSpraak-regel toegepast op een object <i>Persoon</i> . De regel bepaalt de eigenschap <i>leeftijd</i> waarbij de eigenschap <i>geboortedatum</i> wordt gebruikt. Daarnaast gebruikt de regel een datum waarop de leeftijd bepaald moet worden, in dit geval 11 november 2022. ....                                                           | 14  |
| Figuur 2: een objecttype <i>Natuurlijk persoon</i> met twee eigenschappen, de <i>geboortedatum</i> en de <i>leeftijd</i> , dat twee objecten specificeert, oftewel instanties van dit objecttype. De eigenschap <i>geboortedatum</i> heeft bij het aanmaken van de instantie een waarde gekregen, de waarde van de eigenschap <i>leeftijd</i> is nog leeg..... | 16  |
| Figuur 3: drie instanties van objecttype <i>Natuurlijk persoon</i> met allemaal het Boolean attribuut <i>minderjarig</i> . ....                                                                                                                                                                                                                                | 24  |
| Figuur 4: drie instanties van objecttype <i>Natuurlijk persoon</i> waarvan één het kenmerk <i>minderjarig</i> heeft. ....                                                                                                                                                                                                                                      | 24  |
| Figuur 5: een <i>Natuurlijk persoon</i> instantie, met het attribuut <i>inkomen</i> met twee dimensies. ....                                                                                                                                                                                                                                                   | 27  |
| Figuur 6: objecttype <i>Natuurlijk persoon</i> en een instantie ervan. ....                                                                                                                                                                                                                                                                                    | 32  |
| Figuur 7: feittype <i>Vlucht van natuurlijke personen</i> dat een instantie van objecttype <i>Vlucht</i> koppelt aan drie instanties van objecttype <i>Natuurlijk persoon</i> . ....                                                                                                                                                                           | 35  |
| Figuur 8: feittype <i>Vlucht van natuurlijke personen</i> . Het feittype omschrijft de enkelvoudige rol <i>Reis</i> die geïntanceerd wordt door objecttype <i>Vlucht</i> , en de meervoudige rol <i>passagiers</i> die geïntanceerd wordt door objecttype <i>Natuurlijk persoon</i> . ....                                                                     | 35  |
| Figuur 9: wederkerig feittype <i>Partnerrelatie</i> . De enkelvoudige rol <i>partner</i> legt een relatie tussen twee instanties van objecttype <i>Natuurlijk persoon</i> . ....                                                                                                                                                                               | 36  |
| Figuur 10: een eenvoudige onderwerpexpressie die verwijst naar een willekeurige instantie van <i>Natuurlijk persoon</i> . ....                                                                                                                                                                                                                                 | 49  |
| Figuur 11: selectie bij een universeel onderwerp. ....                                                                                                                                                                                                                                                                                                         | 50  |
| Figuur 12: De regel <i>bepaal leeftijd</i> en drie instanties van <i>Natuurlijk persoon</i> . ....                                                                                                                                                                                                                                                             | 51  |
| Figuur 13: de regel <i>bepaal leeftijd</i> wordt drie keer uitgevoerd, elke keer voor een andere instantie van <i>Natuurlijk persoon</i> . ....                                                                                                                                                                                                                | 52  |
| Figuur 14: de regel <i>bepaal leeftijd</i> met het universele onderwerp <i>een Natuurlijk persoon</i> wordt voor de tweede keer uitgevoerd (vandaar aan de linkerzijde geen zwarte stippellijn maar een paarse doorgetrokken lijn). ....                                                                                                                       | 53  |
| Figuur 15: een onderwerpketen. De verwijzing gaat van de instantie van <i>Vlucht</i> , via de rollen <i>reis</i> en <i>passagier</i> uit het feittype <i>Vlucht van Natuurlijke personen</i> naar de relevante instanties van <i>Natuurlijk persoon</i> . ....                                                                                                 | 54  |
| Figuur 16: een ObjectCreatie expressie. Een nieuwe instantie van <i>Contingent treinmiles</i> wordt aangemaakt die met een nieuw feit <i>reis met contingent treinmiles</i> verbonden is met de al bestaande instantie van <i>Vlucht</i> . ....                                                                                                                | 100 |
| Figuur 17: een FeitCreatie expressie. Een instantie van <i>Contingent treinmiles</i> wordt verbonden met een instantie van <i>Natuurlijk persoon</i> via het feittype <i>verdeling contingent treinmiles over passagiers</i> . ....                                                                                                                            | 101 |

## 15 Lijst met tabellen

|                                                                                               |     |
|-----------------------------------------------------------------------------------------------|-----|
| Tabel 1: historie specificatiedocument. ....                                                  | 7   |
| Tabel 2: de vijf standaarddatatypes in RegelSpraak. ....                                      | 22  |
| Tabel 3: voorbeelden van de verschillende afrondingsmethodes. ....                            | 64  |
| Tabel 4: het aantal decimalen van een optelling. ....                                         | 66  |
| Tabel 5: lege waarden bij operator “plus”. ....                                               | 66  |
| Tabel 6: het aantal decimalen van een aftrekking. ....                                        | 67  |
| Tabel 7: lege waarden en aftrekken bij operator “min”. ....                                   | 68  |
| Tabel 8: lege waarden en aftrekken bij operator “verminderd met”. ....                        | 68  |
| Tabel 9: het aantal decimalen bij het vermenigvuldigen. ....                                  | 69  |
| Tabel 10: lege waarden bij operator “maal”. ....                                              | 69  |
| Tabel 11: aantal decimalen en gedeeld door/gedeeld door (ABS). ....                           | 71  |
| Tabel 12: lege waarden bij operator “gedeeld door” (en impliciet “gedeeld door (ABS)”). ....  | 71  |
| Tabel 13: lege waarden en operator “van” (percentage bepalen). ....                           | 73  |
| Tabel 14: lege waarden en operator “tijdsduur van ... tot ...”. ....                          | 75  |
| Tabel 15: lege waarden en datum plus/min tijdseenheid. ....                                   | 76  |
| Tabel 16: de predicaten in RegelSpraak en de datatypes die erbij gebruikt moeten worden. .... | 86  |
| Tabel 17: rekenvoorbeeld bij Verdeling. ....                                                  | 105 |
| Tabel 18: gegevens voor rekenvoorbeeld verdelingen. ....                                      | 107 |
| Tabel 19 Bijzondere waarden bij verdeling. ....                                               | 108 |
| Tabel 20: de gebruikte EBNF symbolen. ....                                                    | 124 |
| Tabel 21: begrippenlijst. ....                                                                | 136 |
| Tabel 22: EBNF symbolen voor de Corba (OMG), W3C, en ISO/IEC 14977 standaarden. ....          | 143 |

## 16 Begrippenlijst

In dit document zijn vanaf hoofdstuk 3 bij het introduceren van een nieuw begrip het eerste voorkomen ervan **vet** weergegeven. Dit hoofdstuk bevat een begrippenlijst met alleen die begrippen die niet in de tekst zijn uitgelegd.

Verder: de woorden in de kolom *Betekenis* die ingesloten zijn met [...] zijn begrippen waarvan een betekenis is gedefinieerd.

| Begrip                | Betekenis                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Complex getal         | Een uitbereiding van [reële getallen], die stelt dat alle getallen op een vlak moeten voorkomen                                                                                                                                                                                                                                                                                                                                                 |
| Decimaal getal        | <i>Synoniem van "Getal met x decimalen"</i>                                                                                                                                                                                                                                                                                                                                                                                                     |
| Geen-gehele getallen  | Getallen met decimalen achter de komma                                                                                                                                                                                                                                                                                                                                                                                                          |
| Geheel getal          | Een mogelijke invulling van een verplichte beperking voor een attribuut met numeriek datatype of percentage datatype die stelt dat alle getallen geschreven moeten worden zonder decimalen achter de komma                                                                                                                                                                                                                                      |
| Getal met x decimalen | Een mogelijke invulling van een verplichte beperking voor een attribuut met numeriek datatype of percentage datatype die stelt dat alle getallen geschreven moeten worden met een x aantal decimalen achter de komma                                                                                                                                                                                                                            |
| Irrationeel getal     | Een irrationaal getal is een reëel getal dat niet een rationaal getal is. Rationale en irrationale getallen samen vormen de reële getallen. Omdat rationale getallen het quotiënt (breuk) zijn van twee gehele getallen, is een irrationaal getal een reëel getal dat niet te schrijven is als een quotiënt van twee gehele getallen. Bekende voorbeelden van irrationale getallen zijn $\sqrt{2}$ (de vierkantswortel uit 2), $\pi$ (pi) en e. |
| Negatief              | Een mogelijke invulling van een optionele beperking voor een attribuut met numeriek datatype of percentage datatype die stelt dat alle getallen kleiner dan 0 moeten zijn                                                                                                                                                                                                                                                                       |
| Positief              | Een mogelijke invulling van een optionele beperking voor een attribuut met numeriek datatype of percentage datatype die stelt dat alle getallen groter dan 0 moeten zijn                                                                                                                                                                                                                                                                        |
| Rationeel getal       | Een [reëel getal] dat kan worden herschreven als de breuk van twee [gehele getallen], waarvan het tweede getal geen nul kan zijn                                                                                                                                                                                                                                                                                                                |
| Reëel getal           | Alle getallen die op een rechte lijn kunnen voorkomen                                                                                                                                                                                                                                                                                                                                                                                           |

Tabel 21: begrippenlijst.



## 17 Index

|                                                            |                       |                               |                                             |
|------------------------------------------------------------|-----------------------|-------------------------------|---------------------------------------------|
| aandeel in een verdeelplafond.....                         | <b>104</b>            | enkele conditie .....         | <b>92</b>                                   |
| aantal aggregatie.....                                     | <b>57</b>             | enkelvoudige expressie.....   | <b>40</b>                                   |
| achterliggende nullen.....                                 | 62                    | enkelvoudige rol .....        | <b>34, 35</b>                               |
| actie .....                                                | <b>38, 39</b>         | enkelvoudige voorwaarde ..... | <b>110</b>                                  |
| afronden .....                                             | <b>63</b>             | enumeratie .....              | <b>23, 48, 125</b>                          |
| afrondpositie .....                                        | <b>63</b>             | expressie .....               | <b>40, 115</b>                              |
| aggregatie.....                                            | <b>56, 57, 58, 60</b> | feitcreatie.....              | 38, <b>101</b>                              |
| ALEF .....                                                 | <b>9, 10</b>          | feittype.....                 | <b>34, 100, 101, 104</b>                    |
| alle .....                                                 | 110                   | gedeeld door.....             | 61, <b>70, 71</b>                           |
| alternatief-symbool.....                                   | <b>123</b>            | gedeeld door (ABS) .....      | 61, <b>70</b>                               |
| attribuut . <b>17, 22, 23, 26, 27, 31, 34, 38, 50, 60,</b> |                       | GegevensSpraak.....           | <b>8</b>                                    |
| 98, 102, 103, 104, 126                                     |                       | geheel getal.....             | 18, 21, 64, 76                              |
| Backus-Naur Form .....                                     | <b>121</b>            | geldigheidsperiode .....      | <b>37</b>                                   |
| basiseenheid.....                                          | <b>28, 29</b>         | gelijkstelling .....          | 38, <b>98, 117</b>                          |
| basispatroon.....                                          | 37, <b>38</b>         | groepering.....               | <b>123</b>                                  |
| beperken .....                                             | <b>18</b>             | herhaling.....                | <b>123</b>                                  |
| beslistabel.....                                           | <b>117</b>            | in dagen .....                | <b>20, 22</b>                               |
| beziëld objecttype .....                                   | <b>16</b>             | in millisecondes .....        | <b>20, 22</b>                               |
| bezittelijk kenmerk.....                                   | <b>25</b>             | inconsistent.....             | <b>102</b>                                  |
| bijvoeglijk kenmerk .....                                  | <b>25</b>             | initialisatie.....            | 38, <b>103</b>                              |
| boolean datatype .....                                     | <b>19, 21, 85</b>     | instantie .....               | <b>15, 16, 31, 34, 38, 55, 99, 100, 102</b> |
| complex getal .....                                        | 18                    | irrationeel getal .....       | 18                                          |
| concatenatie.....                                          | <b>56, 89</b>         | jaar uit.....                 | <b>76</b>                                   |
| conclusie-kolom .....                                      | <b>118</b>            | kenmerk.....                  | <b>24, 26, 39, 90, 102, 103, 126</b>        |
| conditie.....                                              | 102, 110              | kenmerkcheck.....             | <b>90</b>                                   |
| conditie-kolom .....                                       | <b>118</b>            | kenmerktoekenning.....        | 25, 38, 39                                  |
| <b>condities.....</b>                                      | <b>84</b>             | knip .....                    | 41                                          |
| consistentieregel .....                                    | <b>102</b>            | kwantificatie .....           | <b>54, 93, 110</b>                          |
| constructie.....                                           | 38, <b>100</b>        | label .....                   | <b>27, 60</b>                               |
| controleteken .....                                        | <b>124</b>            | lege waarde .....             | <b>17, 38, 59, 103</b>                      |
| dag uit.....                                               | <b>76</b>             | literal expressie.....        | <b>47</b>                                   |
| dagsoort .....                                             | <b>36, 75</b>         | maal .....                    | 61, <b>68, 70</b>                           |
| dagsoortdefinitie .....                                    | <b>36, 38, 108</b>    | maand uit.....                | <b>76</b>                                   |
| datatype .....                                             | 17, 21, 34, 40        | maximale waarde .....         | <b>58, 59</b>                               |
| datum-tijd datatype .....                                  | <b>20, 22, 85</b>     | meervoudige expressie.....    | <b>40, 57</b>                               |
| decimaal getal .....                                       | 18, 21                | meervoudige rol.....          | <b>34</b>                                   |
| definitie .....                                            | <b>123</b>            | meervoudige voorwaarde .....  | <b>110</b>                                  |
| dimensie .....                                             | <b>26, 27, 60</b>     | min .....                     | 61, <b>67, 75</b>                           |
| domein .....                                               | <b>22, 23</b>         | minimale waarde .....         | <b>58, 59</b>                               |
| EBNF .....                                                 | <b>121, 124, 143</b>  | naamgevingconstructie .....   | <b>37, 117</b>                              |
| eenheid.....                                               | 20, <b>28, 66, 67</b> | naamwoord.....                | <b>15, 17</b>                               |
| eenheidssysteem.....                                       | <b>28, 29, 66, 67</b> | naar beneden afronden .....   | <b>63, 64</b>                               |
| eerste paasdag van.....                                    | <b>77</b>             | naar boven afronden .....     | <b>63, 64</b>                               |
| elementaire voorwaarde.....                                | <b>110</b>            | negatief .....                | 18, 21                                      |
| elfproef.....                                              | <b>87</b>             | niet-negatief .....           | <b>18, 21</b>                               |

|                              |                                         |                                                                               |                                             |
|------------------------------|-----------------------------------------|-------------------------------------------------------------------------------|---------------------------------------------|
| niet-terminaal symbool ..... | <b>122</b>                              | samengestelde eenheden.....                                                   | <b>28</b>                                   |
| numeriek .....               | 62, 65                                  | samengestelde voorwaarde .....                                                | 55, <b>110</b>                              |
| numeriek datatype ...        | <b>18</b> , 20, 21, 38, 58, 61, 85, 104 | selectie .....                                                                | <b>50</b> , 56                              |
| object.....                  | <b>14</b> , 15, 100                     | selector .....                                                                | <b>50</b>                                   |
| objecttype .....             | <b>15</b> , 16, 23, 24, 34, 40, 101     | sommatie aggregatie .....                                                     | <b>58</b>                                   |
| objecttypedefinitie .....    | <b>15</b>                               | subselectie .....                                                             | <b>55</b>                                   |
| omrekenfactor.....           | <b>29</b>                               | tekst datatype .....                                                          | <b>19</b> , 21, 48, 85                      |
| omrekenspecificatie .....    | <b>28</b> , <b>29</b>                   | terminaal symbool .....                                                       | <b>122</b>                                  |
| onderwerp.....               | 50                                      | Tijd .....                                                                    | 66, 67                                      |
| onderwerpexpressie.....      | <b>49</b>                               | tijdlijn 17, 26, 30, 31, 34, 40, 41, 43, 44, 45, 46, 47, 78, 79, 81, 113, 126 |                                             |
| onderwerpketen.....          | <b>53</b>                               | tijdsafhankelijk.....                                                         | 30, 38, 40, 41, 43, 78, 80, 91, 95, 96, 113 |
| ontvanger .....              | <b>104</b>                              | tijdsduur van .....                                                           | <b>74</b>                                   |
| optioneel .....              | <b>124</b>                              | tijdsevenredig .....                                                          | 41, 80, 81, 82, 83, 95, 133                 |
| optionele beperking .....    | 20                                      | TOKA-casus .....                                                              | 10, <b>12</b> , 139                         |
| parameter.....               | 28, <b>34</b>                           | uniciteit .....                                                               | <b>88</b>                                   |
| percentage datatype .....    | <b>20</b> , 22, 61, 85                  | universeel onderwerp.....                                                     | 49                                          |
| plus .....                   | 61, 75                                  | <i>Valuta ISO-4217</i> .....                                                  | <b>28</b>                                   |
| positief.....                | 18, 21                                  | variabele .....                                                               | <b>39</b> , 115                             |
| precisie .....               | 62                                      | variabelendeel .....                                                          | 38, 39, 40, <b>115</b>                      |
| predicaat.....               | 55, <b>84</b> , 85, 110                 | verdeelplafond.....                                                           | <b>104</b>                                  |
| rationeel getal .....        | 18, 71, 72                              | verdeler.....                                                                 | <b>104</b>                                  |
| reëel getal.....             | 18, 21                                  | verdeling .....                                                               | 38, <b>104</b>                              |
| RegelSpraak .....            | 10, 11, 13                              | vereniging .....                                                              | <b>89</b>                                   |
| RegelSpraak syntax.....      | 10, <b>121</b>                          | vergelijking.....                                                             | <b>39</b>                                   |
| regelversie .....            | <b>37</b> , 117                         | vergelijkingspredicaat.....                                                   | <b>86</b>                                   |
| rekendatum .....             | 37, <b>48</b>                           | verminderd met.....                                                           | <b>67</b>                                   |
| <i>rekenjaar</i> .....       | <b>48</b>                               | verplichte beperking.....                                                     | 20                                          |
| rekenkundig afronden.....    | <b>64</b>                               | voorwaardendeel.....                                                          | 38, 39, 40, 99, <b>110</b> , 115, 118       |
| resultaatdeel .....          | 38, 39, 40, <b>98</b> , 110, 117, 118   | wederkerig feittype .....                                                     | <b>35</b>                                   |
| richting nul afronden.....   | <b>64</b>                               | weg van nul afronden .....                                                    | <b>64</b>                                   |
| rol .....                    | <b>34</b> , 38, 50, 53, 90, 102, 103    | wortel van .....                                                              | <b>71</b> , <b>72</b>                       |
| rolcheck .....               | <b>90</b>                               | worteltrekken .....                                                           | 61, <b>71</b> , <b>72</b>                   |
| samengestelde conditie ..... | <b>93</b>                               |                                                                               |                                             |

## Bijlage 1: de TOKA-casus

In dit document wordt de TOKA-casus (Kalkema, 2019) gebruikt om het gebruik van de specificaties te illustreren.

### Wet Treinen Op Korte Afstand 2019

#### *Disclaimer*

Voor de cursus is een casus gekozen die enigszins aansluit bij de wereld van de Belastingdienst. De in de casus opgenomen informatie echter volledig verzonnen en heeft geen enkele relatie met eventuele werkelijke dossiers voor belastingwetgeving.

#### **Geldend vanaf 01-01-2019**

##### *Artikel 1*

Onder de naam 'treinen op korte afstand' wordt een belasting geheven ter zake van vluchten.

##### *Artikel 2*

Voor de toepassing van deze wet en de daarop berustende bepalingen wordt verstaan onder:

1. vlucht: een reis met vliegtuig, waarbij het beginpunt wordt verlaten.
2. beginpunt: de plaats waar personen in het vervoermiddel stappen.
3. vliegtuig: vervoermiddel, ingericht voor het vervoeren van personen, waarmee verplaatsing door de lucht mogelijk is.
4. vliegticket: vervoerbewijs dat een persoon recht geeft op vervoer per vliegtuig.

##### *Artikel 3*

Met betrekking tot vluchten wordt de belasting geheven van de vliegtuigmaatschappij voor de op het vliegticket vermelde persoon.

##### *Artikel 4*

1. De belasting moet op aangifte worden voldaan.
2. In afwijking van artikel 10, tweede lid en artikel 19, derde lid, van de Algemene wet inzake rijksbelastingen:
  - a. moet de belasting worden voldaan voordat de vlucht plaatsvindt;
  - b. wordt de betaling gelijktijdig met de aangifte gedaan.

##### *Artikel 5*

1. De belasting ter zake van de vlucht wordt bepaald aan de hand van de volgende tabel.

| Bij een afstand in kilometer s van meer dan | maar niet meer dan | bedraagt de belasting voor een vlucht het in kolom III vermelde bedrag, verminderd met het bedrag dat wordt berekend door het in kolom IV vermelde bedrag te vermenigvuldigen met het aantal kilometers dat de in kolom I vermelde afstand te boven gaat |        |
|---------------------------------------------|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| I                                           | II                 | III                                                                                                                                                                                                                                                      | IV     |
| -                                           | 500                | € 125                                                                                                                                                                                                                                                    | € 0,01 |
| 500                                         | 1000               | € 120                                                                                                                                                                                                                                                    | € 0,03 |
| 1000                                        | -                  | € 0                                                                                                                                                                                                                                                      | € 0    |

2. In afwijking van het eerste lid wordt de belasting ter zake van de vlucht door personen tussen de 18 en 25 jaar, dan wel door personen die de leeftijd van 65 jaar hebben bereikt, bepaald aan de hand van de volgende tabel.

| Bij een afstand in kilometers van meer dan | maar niet meer dan | bedraagt de belasting voor een vlucht het in kolom III vermelde bedrag, verminderd met het bedrag dat wordt berekend door het in kolom IV vermelde bedrag te vermenigvuldigen met het aantal kilometers dat de in kolom I vermelde afstand te boven gaat |        |
|--------------------------------------------|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| I                                          | II                 | III                                                                                                                                                                                                                                                      | IV     |
| -                                          | 500                | € 170                                                                                                                                                                                                                                                    | € 0,02 |
| 500                                        | 1000               | € 160                                                                                                                                                                                                                                                    | € 0,04 |
| 1000                                       | -                  | € 0                                                                                                                                                                                                                                                      | € 0    |

Hierbij dient als leeftijd van een persoon te worden aangehouden de leeftijd van de persoon op de datum van de vlucht.

3. Indien de afstand in kilometers onbekend is dient deze te worden gesteld op 0.
4. Het overeenkomstig het eerste, dan wel het tweede, lid berekende bedrag aan belasting dient te worden vermeerderd met een verhoging bepaald aan de hand van de onderstaande tabel.

| Bij een reisduur per trein in minuten van meer dan | maar niet meer dan | bedraagt de vermeerdering het in kolom III vermelde percentage van het overeenkomstig het eerste, dan wel het tweede, lid berekende bedrag |
|----------------------------------------------------|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| I                                                  | II                 | III                                                                                                                                        |
| -                                                  | 300                | 100 %                                                                                                                                      |
| 300                                                | 600                | 50 %                                                                                                                                       |
| 600                                                | -                  | 0 %                                                                                                                                        |

5. Het bedrag van de belasting op grond van de tabel wordt in geval van een duurzame verplaatsing waarbij gebruik wordt gemaakt van een vliegtuig dat voor minder dan 50 procent op fossiele brandstof vliegt, verminderd met een bedrag van € 10. Deze vermindering geldt uitsluitend voor personen van 65 jaar of ouder op de datum van de verplaatsing en bij een verplaatsing over een afstand die minimaal gelijk aan de afstand die geldt als het maximum van de eerste tariefschijf in lid 2.
6. In afwijking van het in het eerste en tweede lid bepaalde bedrag aan belasting, bedraagt de belasting nihil indien het eindpunt van de vlucht niet bereikbaar is per trein of als er sprake is van een rondvlucht waarbij de luchthavens van vertrek en bestemming dezelfde zijn.
7. In afwijking van het in het eerste en tweede lid bepaalde bedrag aan belasting, bedraagt de belasting nihil indien het bedrag aan belasting negatief is.

8. Voor de toepassing van dit artikel wordt de afstand in kilometers tussen twee plaatsen gemeten op een bij ministeriële regeling te bepalen wijze.

#### Artikel 6

1. In geval van een verhoging van de in artikel 5 opgenomen tarieven wordt voor een vlucht die voorafgaande aan het tijdstip waarop de verhoging in werking treedt, is geboekt, de belasting berekend op de voet van artikel 5 zoals dat luidde voor de inwerkingtreding, tenzij het boeken van de vlucht meer dan drie maanden voor de inwerkingtreding heeft plaatsgevonden. Indien het boeken meer dan drie maanden voor de inwerkingtreding heeft plaatsgevonden, wordt de belasting berekend op de voet van artikel 5, zoals dat luidt nadat de verhoging in werking treedt.
2. In geval van een verlaging van de in artikel 5 opgenomen tarieven wordt voor een vlucht die voorafgaande aan het tijdstip waarop de verlaging in werking treedt, is geboekt, de belasting berekend op de voet van artikel 5, zoals dat luidt nadat de verlaging in werking is getreden.
3. Bij ministeriële regeling kunnen nadere regels worden gesteld ter uitvoering van dit artikel.

#### Artikel 7

- Ter bevordering van het reizen met de trein wordt bij iedere vlucht een contingent Treinmiles verdeeld onder de passagiers. Deze treinmiles zijn een wettig betaalmiddel voor de aanschaf van vervoersbewijzen in het Openbaar Vervoer zoals vastgelegd in de Wet Openbaar Vervoer op korte afstand, art. 3.
- Het te verdelen contingent treinmiles bestaat uit een contingent van 2500 treinmiles per vlucht. Dit aantal treinmiles wordt opgehoogd met een variabel deel op basis van het aantal passagiers van de vlucht. Het variabele deel betreft een contingent van het aantal passagiers maal 100 treinmiles.
- Het beschikbare contingent wordt verdeeld op basis van leeftijd. Treinmiles worden oplopend op leeftijd toegewezen aan passagiers. Passagiers van 64 jaar en jonger krijgen daarbij maximaal 500 treinmiles toegewezen. Passagiers van 65 jaar en ouder krijgen maximaal 250 treinmiles toegewezen.
- Ter bevordering van het reizen met de trein krijgen de passagiers op basis van de provincie waarin zij woonachtig zijn een woonregio-factor. Bij gelijke leeftijd wordt de woonregio-factor gebruikt om het nog beschikbare (deel van het) te verdelen contingent treinmiles naar rato te verdelen.

| Provincie                                        | Woonregio-factor |
|--------------------------------------------------|------------------|
| Noord-Holland , Zuid-Holland, Utrecht            | 3                |
| Noord-Brabant, Gelderland, Overijssel, Flevoland | 2                |
| Friesland, Groningen, Drenthe, Zeeland, Limburg  | 1                |

#### Artikel 8

In afwijking van artikel 3 wordt met betrekking tot rondvluchten geen belasting geheven voor de op het vliegticket vermelde persoon, maar is de luchtvaartmaatschappij ter zake van de rondvlucht, waarbij de luchthavens van vertrek en bestemming dezelfde zijn, een bedrag aan belasting van €259 verschuldigd.

#### *Artikel 9*

- 1) Bij algemene maatregel van bestuur kunnen ter verzekering van een juiste toepassing van de wet nadere regels worden gesteld ter aanvulling van de in deze wet geregelde onderwerpen.
- 2) Bij ministeriële regeling kunnen nadere regels worden gesteld ten behoeve van de uitvoering van de in het eerste lid bedoelde algemene maatregel van bestuur.

#### *Artikel 10*

De inwerkingtreding van deze wet wordt bij koninklijk besluit geregeld.

#### *Artikel 11*

Deze wet kan worden aangehaald als Wet treinen op korte afstand 2018.

## Bijlage 2: de EBNF alfabetten van CORBA(OMG), W3C en ISO/IEC 14977

Om een idee te krijgen van de verschillen tussen de eerder genoemde EBNF versies staan hieronder in tabelvorm de alfabetten van drie versies tegen elkaar uitgezet. In de laatste kolom staat daarbij de functie van de gebruikte symbolen benoemd. Wanneer een cel in de tabel leeg is gelaten betekent dit dat de variant deze functie niet biedt.

| Corba (OMG) | W3C                             | ISO/IEC 14977 | Functie                                                      |
|-------------|---------------------------------|---------------|--------------------------------------------------------------|
| ::=         | ::=                             | =             | Is gedefinieerd als                                          |
|             |                                 |               | Alternatief                                                  |
| <tekst>     | tekst                           | tekst         | Variabele                                                    |
| "tekst"     | "tekst"                         | "tekst"       | Karakterreeks                                                |
| Spatie      | Spatie                          | ,             | concatenatie                                                 |
|             |                                 | -             | uitzondering                                                 |
| ...*        | *...                            | (...)         | Mag 0 of meer keer voorkomen                                 |
| ...+        | 1*...                           | (...)-        | Mag 1 of meer keer voorkomen                                 |
|             | X*...                           | 3 * ...       | Mag X of meer keer voorkomen                                 |
|             | *Y...                           |               | Mag 0 t/m Y keer voorkomen                                   |
|             | X*Y...                          |               | Mag X t/m Y keer voorkomen                                   |
|             | X...                            |               | Komt precies X keer voor                                     |
|             | X#Y(...)<br>(... X*Y(", " ...)) |               | door komma's gescheiden lijst met tussen de X en Y elementen |
| ()          | ()                              | ()            | Groepering                                                   |
| []          | [...]<br>*1(...)                | []            | Optioneel - Ingesloten kan 0 of 1 keer voorkomen             |
| #           | ;...                            | (*...*)       | Commentaar                                                   |
|             |                                 |               |                                                              |
| \n          | \n                              | \n            | New line/line feed                                           |
| \t          | \t                              | \t            | Horizontal tab                                               |
| \v          |                                 | \v            | Vertical tab                                                 |
| \b          | \b                              | \b            | Backspace                                                    |
| \r          | \r                              | \r            | Carriage return                                              |
| \f          | \f                              | \f            | Form feed                                                    |
| \a          |                                 | \a            | Alert                                                        |
| \\          |                                 | \\            | Backslash                                                    |
| \?          |                                 |               | Vraagteken                                                   |
| \'          |                                 |               | Single quote                                                 |
| \"          | \"                              |               | Double quote                                                 |

Tabel 22: EBNF symbolen voor de Corba (OMG), W3C, en ISO/IEC 14977 standaarden.

## Bijlage 3: basisstructuur RegelSpraak in 3 EBNF varianten

Hieronder is de basisstructuur van RegelSpraak als voorbeeld genomen om de drie BNF varianten te vergelijken:

### CORBA

```
<regel> ::= <regelnaam> \n <regelversie> {\n <regelversie>}*
<regelnaam> ::= "Regel" <karakterreeks>
<regelversie> ::= <versie> \n <regelsoort> \n [<voorwaarde>]
<versie> ::= \t "geldig" ("altijd" | "vanaf" {<jaaartal> | <datum>} | "t/m" {<jaaartal> | <datum>} | "vanaf" {<jaaartal> | <datum>} "t/m" {<jaaartal> | <datum>})
```

### W3C

```
regel ::= regelnaam \n regelversie *(\n regelversie)
regelnaam ::= "Regel" karakterreeks
regelversie ::= versie \n regelsoort \n [voorwaarde]
versie ::= \t "geldig" ("altijd" | "vanaf" (jaaartal | datum) | "t/m" (jaaartal | datum) | "vanaf" (jaaartal | datum) "t/m" (jaaartal | <datum>))
```

### ISO/IEC 14977

```
regel = regelnaam, \n, regelversie, (\n, regelversie)
regelnaam = "Regel", karakterreeks
regelversie = versie, \n, regelsoort, \n, [voorwaarde]
versie = \t, "geldig", ("altijd" | "vanaf", (jaaartal | datum) | "t/m", (jaaartal | datum) | "vanaf", (jaaartal | datum), "t/m", (jaaartal | <datum>))
```