



Australian Government
**Australian Institute of
Health and Welfare**

Community Mental Health Care

National Minimum Data Set

2024-25 version 6.00

The metadata for CMHC 6.00 can be found on the [Online Validator metadata page](#)

As at 1 October, 2024

Table of Contents

- 1. 2024-25 CMHC NMDS.....3
 - 1.1. Essential definitions3
 - 1.2. Changes for 2024-25.....3
 - 1.3. Reporting service provider entities.....3
 - 1.4. Data model of the CMHC Extract.....6
 - 1.5. Data integrity8
 - 1.6. Data set specification (DSS)8
- 2. Submission and validation of CMHC NMDS data files 21
 - 2.1. Timelines 21
 - 2.2. File type and naming convention 21
 - 2.3. Validation 22
 - 2.4. Submission report..... 23
 - 2.5. Additional information and queries 23
- 3. Appendix A: Notes on reporting service contacts for non-uniquely identifiable clients..... 24
- 4. Appendix B: Rules and Virtual Elements 28

1. 2024-25 CMHC NMDS

1.1. Essential definitions

Scope - Community mental health care NMDS 2024-25 (METEOR ID [775620](#)).

Statistical unit - Mental health service contact (METEOR ID [727358](#)).

1.2. Changes for 2024-25

The specific detailed changes to the 2024-25 (version 6.00) specifications, compared to 2023-24 (version 5.51) are listed below.

1.2.1. Changes to the data model

The data model changes to the 2024-25 specifications, compared to 2023-24 are listed in [Table 1.1](#).

Table 1.1 Changes made to 2024-25 data model compared to 2023-24 model

Data items	Details	Rationale
Person—sex, code X	This item is no longer Mandatory and is collected on a Conditional basis with the element Person—Gender, code X. Data must be reported for at least one of the two elements, either Sex or Gender. Data may be reported for both.	To support the implementation of the Gender data element in line with the Admitted Patient Care NMDS.
Person—gender, code X	This item has been added to the NMDS. This data element is collected on a Conditional basis with the element Person—sex, code X. Data must be reported for at least one of the two elements, either Sex or Gender. Data may be reported for both.	To support the implementation of the Gender data element in line with the Admitted Patient Care NMDS.

1.2.2. Changes to definitions

No definitional changes have been made.

1.3. Reporting service provider entities

The reporting of service entities aims to create relationships between the mental health NMDSs, and where possible, the National Outcomes and Casemix Collection (NOCC), Local Hospital Networks/Public Hospital Establishments (PHE) NMDS and Admitted Patient Care (APC) NMDS (see [Table 1.2](#)).

The identifiers used in the CMHC NMDS are:

- State or territory (1 character)
- Region (2 characters)
- Specialised mental health service (4 characters)

- Service unit cluster (5 characters)
- Service unit (6 characters)

Table 1.2 Reporting requirements for mental health and related NMDSs

Identifier element names	METEOR identifier	Community mental health care NMDS	Residential mental health care NMDS	Mental health establishments NMDS	Public hospital establishments NMDS	Admitted patient care NMDS
Australian State or Territory identifier	720081	Yes	Yes	Yes	Yes	Yes
Region identifier	269940	Yes	Yes	Yes	Yes	Yes
Specialised mental health service organisation identifier	404186	Yes	Yes	Yes	No	No
Hospital/ Service unit cluster identifier	722233 (MHE) / 404858 (MHE, CMHC & RMHC)	Yes	Yes	Yes	Yes	Yes
Service unit identifier	721740 (MHE) / 750360 (MHE, CMHC) / 722711 (MHE, RMHC)	Yes	Yes	Yes	No	No

The use of identical identifiers between the various mental health data sets is tested via the Mental Health Establishments Skeleton file (SKL), handled by the Online Validator. The reports section of the CMHC submission will highlight any mismatches which should be rectified either through re-supply of the Skeleton file, or adjustment to the CMHC submission.

The following section explores in more detail the reporting levels used in the CMHC NMDS.

1.3.1. State or territory

This level refers to the state or territory and should be reported using the *State/Territory identifier* data element.

1.3.2. Region

The region refers to an administrative concept not a geographical one. States and territories may have one or more regions into which the state or territory is divided and to which its mental health service organisations belong. Region would be reported using the *Region identifier* (RegId) data element. In the smaller states or in the territories there may only be one or no region. In these cases, the *Region identifier* is to be reported as '00' and the *Region name* (RegName) would repeat the name of the state or territory.

1.3.3. Organisation

The organisation is defined according to the Object class *Specialised mental health service organisation* (METEOR identifier 286449) and reported using the *Organisation Identifier* (OrgId) data element. An organisation is a separately constituted specialised mental health service that is responsible for the clinical governance, administration and financial management of service units providing specialised mental health care. An organisation may consist of one or more service units based in different locations.

1.3.4. Service unit cluster

A specialised mental health service organisation may consist of one or more clusters of service units providing services in admitted patient, residential and ambulatory settings. For example, a specialised mental health service organisation may consist of several hospitals (clusters of admitted patient service units) and/or two or more ambulatory or residential service clusters (for example, a cluster of child and adolescent ambulatory service units, and a cluster of aged residential service units).

To allow service units (as defined below) to be reported individually, but still to be identified as part of a cluster, a separate reporting level has been created called 'Service unit cluster'.

Ambulatory service units will not necessarily belong to a 'cluster'. However, for some ambulatory service units, the cluster the service unit belongs to may be a hospital that contains both an admitted patient and an ambulatory service unit. In this instance the *Service unit cluster identifier* for ambulatory service unit would be the *Hospital identifier*. Other groups of ambulatory service units could also be usefully reported as clusters. For example, clusters may exist of groups of ambulatory services for children and adolescents in particular geographical areas. However, where there is no requirement for a service unit cluster, then all service units within the organisation should be identified under a *Service unit cluster identifier* reported as '00000' and the *Service Unit Cluster name* would use the relevant organisation name.

1.3.5. Service units

The reporting of service units is at the discretion of states and territories. Service units with differing target populations must be reported separately. For example, if a service unit cluster or organisation provided two or more child and adolescent ambulatory service units, jurisdictions have discretion on whether these are reported as one combined child and adolescent ambulatory service unit, or reported as multiple individual service units.

However, identification of service units should not combine target populations—there is no code available to identify 'mixed' target populations. Therefore, where a service entity provides discrete and specifically funded programs for multiple target populations, each of these should be identified as a separate service unit.

Comparison with the MHE NMDS is undertaken using the SKL file. At a minimum, the combination of Organisation and Target population must be the same between the CMHC and MHE submissions. For example, multiple Child and adolescent services within an organisation may be reported in the CMHC file, however, may be rolled together for the MHE submission, and vice versa.

1.3.6. Sector

Sector is not considered part of the identifier. Within this NMDS, sector is an attribute of each service unit.

1.3.7. Consistency of identifiers across reference periods

Where no major service reorganisations have occurred, the region, organisation, service unit cluster and service unit identifiers (RegId, OrgId, ClusId, SUIId) used by a jurisdiction should be preferably identical to the previous year. However, given that all jurisdictions have committed to aligning ID numbers between the different NMDS's, changes in ID numbers due to this process will be accepted, as will be the case for those jurisdictions that have undergone significant reorganisation of service delivery that warrant new service entity identifiers. In these cases, jurisdictions should provide a supplementary mapping document that clearly illustrates the changes in ID numbers between collection periods, at all levels.

Region, organisation, service unit cluster and service unit name changes are acceptable, especially if the new name is more locally relevant. These will be identified as a change in the Online Validator, however will not affect the generation of the historical trends reports for CMHC in future.

1.4. Data model of the CMHC Extract

The basic design of the extract consists of a single data record for each *Mental health service contact*. Each Mental Health Service Contact 'belongs' to a *Person* (the patient or consumer of services), who in turn is linked to a *Community mental health service unit* (the provider of services), which may be linked to a *Community mental health service unit cluster*, which is linked to a *Specialised mental health service organisation*, which is linked to a *Region* which is linked to a *State/Territory*.

The structure of the data to be reported is represented in the data model shown in [Fig. 1.1](#). In the model, a single *Community mental health service unit* has associated records for one or more *Persons*, who each may have one or more *Mental health service contacts*. Each of the seven data model building blocks (state/territory, region, organisation, service unit cluster, service unit, person, contacts) has a unique set of attributes which comprise the NMDS data elements and additional supplementary information.

It should be noted that non-volatile person-level data in respect of patients (Date of Birth, Sex, Country of Birth, Indigenous Status) are separated from person-level data items that may change between service contacts.

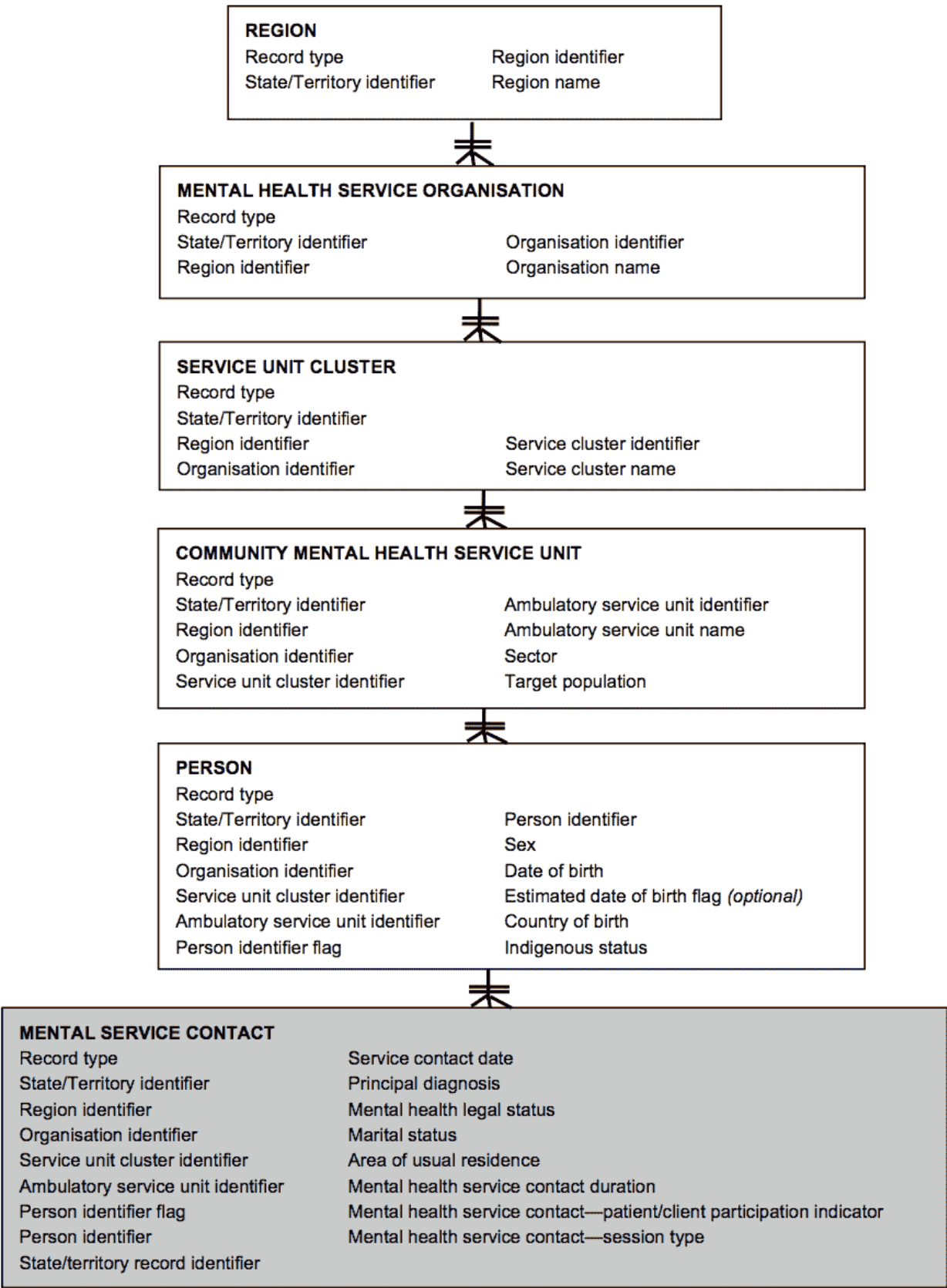


Fig. 1.1 Data model underlying the Community Mental Health Care NMDS data extract

1.5. Data integrity

For cases of missing data (that is, unknown, not stated or not available):

- For **Numeric [Num] fields**, the data should be reported as zero, using leading zeros when necessary to pad out the field to the required length. The principle here is that all numeric fields require a valid value.
- For **Text [Char] fields**, the data should be space-filled to the required length. For single character fields where a 'missing/not stated' value has been specified for a particular data element (for example, '9' has been specified for missing data), use the stated value for 'missing/not stated' rather than simply space filling.

Values in **Date [Date]** fields must be recorded in compliance with the standard format used across the *National health data dictionary*; specifically, dates must be of fixed 8 column width in the format DDMMYYYY, with leading zeros used when necessary to pad out a value. For instance, 13 March 2025 would appear as 13032025.

Values in **Numeric [Num]** fields must be zero-filled and right-justified. These should consist only of the numerals 0 to 9 and the decimal (".") point if applicable to the data element.

Note: Fields defined as 'Numeric' are those that have numeric properties—that is, the values, for example, can be added or subtracted in a manner that is valid. Where a field uses numeric characters that do not have these properties (for example, the use of numbers for *Patient identifier*), the field is defined as 'Character'.

Values in **Character [Char]** fields must be left justified and space-filled. These should consist of any of the printable ASCII character set (that is, excluding control codes such as newline, bell and linefeed).

1.6. Data set specification (DSS)

The following tables specify the order in which the data items should be provided to the AIHW.

The extract format consists of a set of hierarchically ordered *Data records*, of which there are six types (see [Table 1.3](#)):

- Region details records
- Organisation details records
- Service unit cluster details records
- Service unit details records
- Person details records
- Service contact records

In each extract file for any given period, the *Data records* must be preceded by a single *File Header Record* having the structure outlined below in [Table 1.4](#).

All records presented in the extract file should be grouped in the following order: Header Record, Region details records, Organisation details records, Service unit cluster details records, Service unit details records, Person details records, followed by Service contact records.

With the exception of Region, Organisation and Service unit cluster details records, all *Data records* should include the following elements in the order shown:

- Record Type
- Establishment identifier (comprising: *State/Territory identifier*, *Region identifier*, *Organisation identifier*, *Service unit cluster identifier* and *Service unit identifier*)
- Specific data in the format specified for the given record type.

The order of fields in a record must be the same as the order they are specified in the Record Layouts below. Field values should be formatted as shown in the Record Layouts.

The first field in each record must be *Record Type*. Valid values are shown in [Table 1.3](#).

Table 1.3 Valid values for Record Type

Record Type	Description
HR	File Header Record
REG	Region details records
ORG	Organisation details records
CLUS	Service unit cluster details records
SERV	Service unit details records
PER	Person details records
CON	Service contact details records

1.6.1. File header record

The first record of the extract file must be a File Header Record (*Record Type* = 'HR'), and it must be the only such record in the file.

The File Header Record is a quality control mechanism, which uniquely identifies each file that is sent to the AIHW (that is, who sent the file, what date the file was sent, batch number of file, etc). The information contained in the header fields will be checked against the actual details of the file to ensure that the file received has not been corrupted.

The layout of the File Header Record is shown in [Table 1.4](#).

Table 1.4 Record Layout for File Header record within the data extract

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = HR
State/Territory Identifier (State) ¹	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Batch Number (BatchNo)	Char[9]	10	—	Represents the YYYNNNNNN component of the extract file name.
Report Period Start Date (RepStart)	Date[8]	19	—	Report period start date
Report Period End Date (RepEnd)	Date[8]	27	—	Report period end date
Data File Generation Date (GenDt)	Date[8]	35	—	Data file generation date
Data File Type (FileType)	Char[4]	43	—	Value = CMHC
CMHC Specification Version Number (SpecVer)	Char[5]	47	—	Value = 06.00

Record length = 51

Notes

[1]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

1.6.2. Region data record

The extract format for the *Data records* is specified in detail in tables [Table 1.4](#) to [Table 1.10](#). The order of fields in each record must be the same as the order they are shown below. Field values should be formatted as specified.

Table 1.5 Data record layout - Region details

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = REG
State/ Territory Identifier (State) ²	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Region Name (RegName)	Char[60]	12	407187	Common name used to identify the Region.

Record length = 71

Notes

[2]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

1.6.3. Organisation data record

Table 1.6 Data record layout - Organisation details

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = ORG

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
State/ Territory Identifier (State) ³	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Identifier (OrgId)	Char[4]	12	404186	AAAA: Mental health service organisation identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Name (OrgName)	Char[100]	16	405767	Common name used to identify the Organisation

Record length = 115

Notes

[3]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

1.6.4. Service Unit Cluster data record

Table 1.7 Data record layout - Service Unit Cluster

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = CLUS

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
State/ Territory Identifier (State) ⁴	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Identifier (OrgId)	Char[4]	12	404186	AAAA: Mental health service organisation identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Service Unit Cluster Identifier (ClusId)	Char[5]	16	404858	AAAAA: An identifier to indicate that a service unit is one of a cluster of service units, defined through administrative or clinical governance arrangements. If no cluster applies, set to 00000. As this field enables linking with the NMDS for Mental Health Establishments, the identifiers used in this collection should be the same.
Service Unit Cluster Name (ClusName)	Char[100]	21	409209	Common name used to identify the service unit cluster. If no cluster applies, enter organisation name as appears in previous line.

Record length = 120

Notes

[4]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

1.6.5. Service Unit data record

Table 1.8 Data record layout — Service Unit Details

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = <i>SERV</i>

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
State/ Territory Identifier (State) ⁵	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Identifier (OrgId)	Char[4]	12	404186	AAAA: Mental health service organisation identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Service Unit Cluster Identifier (ClusId)	Char[5]	16	404858	AAAAA: An identifier to indicate that a service unit is one of a cluster of service units, defined through administrative or clinical governance arrangements. If no cluster applies, set to 00000. As this field enables linking with the NMDS for Mental Health Establishments, the identifiers used in this collection should be the same.
Ambulatory Service Unit Identifier (SUIId)	Char[6]	21	750360	AAAAAA: Service unit identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Ambulatory Service Unit Name (SUName)	Char[100]	27	750374	Common name used to identify the service unit.
Sector (Sector)	Char[1]	127	269977	1: Public 2: Private

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Target Population (TargetPop) ⁶	Char[1]	128	682403	1: Child and adolescent 2: Older person 3: Forensic 4: General 5: Youth

Record length = 128

Notes

[5]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

[6]

METEOR includes code 7 and 9, but these are not applicable to the CMHC NMDS.

1.6.6. Person Details data record

Attention

Where multiple values of Sex, Gender, Date of birth, Country of birth, or Indigenous status are recorded for different service contacts for one PersId, data providers should adopt the value recorded for the last valid service contact.

Table 1.9 Data record layout — Person Details

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = PER

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
State/ Territory Identifier (State) ⁷	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Identifier (OrgId)	Char[4]	12	404186	AAAA: Mental health service organisation identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Service Unit Cluster Identifier (ClusId)	Char[5]	16	404858	AAAAA: An identifier to indicate that a service unit is one of a cluster of service units, defined through administrative or clinical governance arrangements. If no cluster applies, set to 00000. As this field enables linking with the NMDS for Mental Health Establishments, the identifiers used in this collection should be the same.
Ambulatory Service Unit Identifier (SUIId)	Char[6]	21	750360	AAAAAA: Service unit identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Person Identifier Flag (PersIdFlag) ⁸	Char[1]	27	493279	1: Yes, Patient identifier is for a uniquely identifiable person. 2: No, Patient identifier is for a non-uniquely identifiable person
Person Identifier (PersId)	Char[20]	28	290046	Person identifier is unique and stable for each individual patient within each service unit. Individual service units or collection authorities may use their own alphabetic, numeric or alphanumeric coding systems.

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Sex (Sex) ⁹	Char[1]	48	741686	1: Male 2: Female 3: Another term 9: Not stated / inadequately described
Date of Birth (DoB)	Date[8]	49	287007	The date of birth of the person.
Estimated Date of Birth Flag (DoBFlag) ¹⁰	Char[1]	57	—	1: Date of birth is accurate 2: Date of birth is an estimate 8: Date of birth is a “dummy” date (ie, 09099999) 9: Accuracy of stated date of birth is not known
Country of Birth (CoB)	Char[4]	58	659454	To be provided in accordance with the Standard Australian Classification of Countries (SACC). ABS catalogue no. 1269.0 (2016). Values from 1601-1607, inclusive, are not permitted in this NMDS (Antarctica).
Indigenous Status (IndigSt)	Char[1]	62	602543	1: Aboriginal but not Torres Strait Islander origin 2: Torres Strait Islander but not Aboriginal origin 3: Both Aboriginal and Torres Strait Islander origin 4: Neither Aboriginal nor Torres Strait Islander origin 9: Not stated/inadequately described
Gender (Gender) ¹¹	Char[1]	63	741842	1: Man, or boy, or male 2: Woman, or girl or female 3: Non-binary 4: Different term 5: Prefer not to answer 9: Not stated / inadequately described

Record length = 63

Notes

[7]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

[8]

See Appendix A for further details on unregistered client service contacts.

[9]

Sex and Gender are conditional data elements. Data must be reported for at least one of the two elements, either Sex or Gender. Data may be reported for both.

[10]

Optional data element providing additional information regarding the quality of date of birth data. Code 1 should be used when it is known that the reported date of birth is accurate, code 2 when it is known that one or more parts of the date of birth is an estimate, code 8 when birth date is unknown and a 'dummy' date of birth has been used (that is, 09099999), and code 9 when it is not known whether the date of birth is accurate or an estimate.

[11]

Sex and Gender are conditional data elements. Data must be reported for at least one of the two elements, either Sex or Gender. Data may be reported for both.

1.6.7. Service Contact data record

Table 1.10 Data record layout — Contact Details

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Record Type (RecType)	Char[8]	1	—	Value = CON
State/Territory Identifier (State) ¹²	Char[1]	9	720081	1: New South Wales 2: Victoria 3: Queensland 4: South Australia 5: Western Australia 6: Tasmania 7: Northern Territory 8: Australian Capital Territory
Region Identifier (RegId)	Char[2]	10	269940	AA: Region (values as specified by individual jurisdiction) Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Organisation Identifier (OrgId)	Char[4]	12	404186	AAAA: Mental health service organisation identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Service Unit Cluster Identifier (ClusId)	Char[5]	16	404858	AAAAA: An identifier to indicate that a service unit is one of a cluster of service units, defined through administrative or clinical governance arrangements. If no cluster applies, set to 00000. As this field enables linking with the NMDS for Mental Health Establishments, the identifiers used in this collection should be the same.
Ambulatory Service Unit Identifier (SUIId)	Char[6]	21	750360	AAAAAA: Service unit identifier. Identifiers used in this collection should map to the identifiers used in the NMDS for Mental Health Establishments.
Person Identifier Flag (PersIdFlag) ¹³	Char[1]	27	493279	<p>1: Yes, Patient identifier is for a uniquely identifiable person.</p> <p>2: No, Patient identifier is for a non-uniquely identifiable person</p>
Person Identifier (PersId)	Char[20]	28	290046	Person identifier is unique and stable for each individual patient within each service unit. Individual service units or collection authorities may use their own alphabetic, numeric or alphanumeric coding systems.
State/Territory Record Identifier (RecordId)	Char[10]	48	—	This should be a stable number in the data collection of the jurisdiction.
Service Contact Date (ContDt)	Date[8]	58	737299	The date of each service contact between a health service provider and patient/client.
Principal Diagnosis (DxPrinc)	Char[6]	66	746665	<p>Represented as ANN{.N[N]}</p> <p>The diagnosis established after study to be chiefly responsible for occasioning a mental health service contact, as represented by a code. The principal diagnosis must be a valid code from the International Statistical Classification of Diseases and Related Health Problems, 10th Revision, Australian Modification (ICD-10-AM) (12th Edition) or from the ICD-10-AM Mental Health Manual: An integrated classification and diagnostic tool for community based mental health services (1st Edition).</p>
Mental Health Legal Status (LegalSt)	Char[1]	72	727343	<p>1: Involuntary patient</p> <p>2: Voluntary patient</p> <p>9: Not reported/Unknown</p>

Data Element (Field Name)	Type [Length]	Start	METEOR Identifier	Notes / Values
Marital Status (MaritalSt)	Char[1]	73	766507	1: Never married 2: Widowed 3: Divorced 4: Separated 5: Married (registered and de facto) 6: Not stated/inadequately described
Area of Usual Residence (ResArea)	Char[9]	74	747315	Statistical Area Level 2 (SA2) code (ASGS Edition 3) N(9)
Mental Health Service Contact Duration (ContDur)	Number[3]	83	737218	Valid time measured in minutes. Expressed as NNN
Mental Health Service Contact- Patient/Client Participation Indicator (ContPartic)	Char[1]	86	737291	1: Yes 2: No 8: Unknown
Mental Health Service Contact- Session Type (ContSessType)	Char[1]	87	737307	1: Individual session 2: Group session 8: Unknown

Record length = 87

Notes

[12]

METEOR includes code 9, but that is not applicable to the CMHC NMDS

[13]

See Appendix A for further details on unregistered client service contacts.

2. Submission and validation of CMHC NMDS data files

Submission, delivery and validation of the 2024-25 CMHC data (version 6.00) will occur through the [Online Validator](#). AIHW and the Department of Health will obtain jurisdictional DAT files directly from the Online Validator using the download functionality available to reviewers.

2.1. Timelines

Jurisdictions are requested to propose a file using the Online Validator by **2 January 2026**. The Australian Institute of Health and Welfare (AIHW) and the Department of Health and Aged Care are aiming to have fully reviewed and validated the file by **17 April 2026** to facilitate timelier reporting of CMHC data, in accordance with the schedule in [Table 2.1](#).

Table 2.1 CMHC and RMHC NMDS 2024-25 data validation

Progress point description	Responsibility	Completion Time	Completion date
Compliant file proposed for review	Jurisdictions		2 January 2026
Submitter comment on issues within the issue list	Jurisdictions	7 weeks	20 February 2026
Reviewer reply to issues within the issue list and raise other issues based on historical reports	AIHW	3 weeks	13 March 2026
Resolution of any remaining issues - validation process completed and submission finalised	AIHW/Jurisdiction discussion back and forth	5 weeks	17 April 2026

2.2. File type and naming convention

DAT files should be a single Fixed Format data file, with each record in the file being terminated with Carriage Return (CR) and Line Feed (LF) characters.

The data file will have the naming convention of *CMHCSSSYYYNNNNN.DAT* where:

- CMHC denotes 'Community Mental Health Care'
- SSS is the abbreviation for the State name, using the following convention:
 - New South Wales = NSW
 - Victoria = VIC
 - Queensland = QLD
 - Western Australia = WAU
 - South Australia = SAU
 - Tasmania = TAS
 - Australian Capital Territory = ACT
 - Northern Territory = NTE

- YYYY indicates the reporting year covered in the file, using the convention where financial years are abbreviated by referring to the last calendar year of the pair (for example, 2024-25 is identified as 2025)
- NNNNN represents an incremental batch number (leading zeros present).

Any resubmitted files should have a batch number greater than the file they replace. For example, the first CMHC data file submitted by the Australian Capital Territory covering the 2024-25 year would be named 'CMHCACT202500001.DAT'.

2.3. Validation

Mental Health National Minimum Dataset (NMDS) validation is the process of reviewing and cleaning the mental health service data received from state and territory governments using the [Online Validator](#) a web based validation tool. The process includes:

- Checks on file upload, before the submitter chooses to propose the file for review
- The option for reviewers to accept or reject the proposed file for review
- Collaborative review and validation
 - Review and accept all issues in the issues list in consultation with jurisdictions as necessary.
 - Review data set reports (CMHC and RMHC) and raise additional issues with the jurisdiction as necessary.
- A finalisation step, indicating that the submission process for that reporting cycle is complete.

2.3.1. Checks on file upload

The pre-submission checks ensure that the submitted data file structure is correct: that the data is in the correct layout, that there are no disallowed blank fields and no invalid characters. These checks ensure that each line of data is correctly formatted and aligns to the specifications.

2.3.2. Collaborative review and validation

Collaborative review and validation is the process of reviewing unusual trends in the files. Each collection is different and has its own structure, but the general process of validating is the same. In this process:

1. Submitters work through the issues list, either providing comments on known issues or updating and resubmitting the file. It is expected that most jurisdictions will need to submit updated files multiple times before validation is finalised. Note that jurisdictions must submit comments on issues before the AIHW is able to accept an issue.
2. Submitters and reviewers collaborate, via the Online Validator, when more information is required on an issue. Each jurisdiction will have an AIHW staff member assigned as the lead validator, who will be in touch early in the validation period. Please ensure that any email communication is also Cc'd to the Mentalhealth@aihw.gov.au inbox in case of staff absences or changes in AIHW staffing.

3. The AIHW will review the 'data set reports' and will raise additional queries about any large fluctuations evident in these reports after investigating which regions, organisations and service units are causing them. The AIHW will endeavour to ensure that additional queries are limited to essential issues that impact the state-wide result.

2.4. Submission report

States and territories must complete the submission report, formerly the 'data quality survey', in the Online Validator for the CMHC submission. Details will be used to inform the Quality statements published on METEOR for each NMDS collection period.

2.5. Additional information and queries

AIHW and Logicly staff are available to answer any queries regarding the submission and validation of the 2024-25 CMHC NMDS. In order to obtain a coordinated response, requests should be sent to the following parties simultaneously:

AIHW

Email: mentalhealth@aihw.gov.au

Logicly

Email: support@validator.com.au

3. Appendix A: Notes on reporting service contacts for non-uniquely identifiable clients

3.1. Background

The Community Mental Health Care (CMHC) NMDS is a national data collection that is built from mental health service contact records reported by each State and Territory jurisdiction. The basic concept underpinning the collection is a simple one, involving the following components:

- each service contact recorded for an identified client is reported to the national data pool;
- each service contact is attributable to an individual client by virtue of an anonymous unique patient identifier attached to the service contact record.

This allows each service contact to be linked to an individual client to whom it 'belongs', and provides a basis for reporting on the number of clients in receipt of community mental health care and the services that they receive.

However, since its inception in 2000-01, the data reported by jurisdictions for the CMHC NMDS have included a variable number of service contacts for non-uniquely identifiable clients. For the purposes of this current data extract specification, these are defined in the following terms:

Service contacts for non-uniquely identifiable clients refer to those mental health service contacts for which a person identifier was not recorded.

Typically, when such contacts are reported to the national data set, the person identifier number is not reported but the contact record includes details on one or more person-level data elements (for example, date of birth, sex, country of birth etc).

For some jurisdictions such contacts comprise up to 20% of total service contact records. For a significant number of such records, person-level data elements are also reported. It is evident that these contacts are genuine service contacts in respect of an individual client rather than 'non patient care' activities of community-based mental health services. In this respect, they are fully compliant with the scope of the CMHC NMDS. The only problem is that they have no unique patient identifier attached to the contact record.

A variety of factors underlie the reporting of service contacts for non-uniquely identifiable clients. These include:

- The service provider clinician may record a service contact for a person who has not been assigned a local patient identifier, and for whom there is no intention to assign such an identifier.
- The clinician may record a service contact for a person prior to a local patient identifier being assigned to that individual.
- Alternatively, a clinician may record a service contact for a person for whom a local patient identifier has been assigned but, due to technical limitations, the identifier is not included in the data submitted to the relevant State or Territory health department.

Differences between jurisdictions in how unregistered contact data are handled also contribute to the variation that is evident in the national data set. Three key differences that affect the national collection are:

- *Differences in patient registration guidelines*—jurisdictions differ in the extent to which policy documentation has been developed to guide local agencies on when to assign patient identifiers in community settings. Some jurisdictions have prepared detailed instructions for agencies that specify the requirement for patient identifiers to be assigned to all clients for whom a clinically significant service is provided, regardless of whether the client is accepted for ongoing services. For others, decisions about when to assign identifiers are left to the discretion of local agencies.
- *Different approaches to whether service contacts for non-uniquely identifiable clients are held in State and Territory-level collections*—while some jurisdictions require such contacts to be reported to the State/Territory health department, others do not, or have no capacity to store or process such data.
- *Different approaches by states and territories to whether service contacts for non-uniquely identifiable clients are included in data submitted for the CMHC NMDS*—for those jurisdictions that hold such contact data within the State- or Territory-level collections, differences exist as to whether such records are included in the data submitted for National Minimum Data Set purposes.

3.2. How contacts for non-uniquely identifiable clients should be reported for CMHC extract purposes

Contacts for non-uniquely identifiable clients create an anomaly for the national analysis and reporting of data submitted by states and territories. The following approach to the reporting of non-uniquely identifiable client contacts has been incorporated in the extract specifications to promote consistency where these contacts are reported:

- States and territories that choose to report service contacts for non-uniquely identifiable clients to the CMHC NMDS may continue to do so, following the requirements of the extract specification. There is no expectation that jurisdictions that have not previously submitted such data will do so in the future—the choice is entirely the responsibility of each State and Territory.
- Where such records are submitted to the national collection, they will conform to the data model underpinning the extract—that is, for each unregistered client contact record, there must be a matching *Person details* record.
- Service contacts for non-uniquely identifiable clients will be clearly differentiated through the addition of a *Person identifier flag* on all *Service contacts* and *Person details* records.
- For each *Service contact details* record that represents a contact with a non-uniquely identifiable client, a unique patient identifier should be created to fill the *Patient identifier field*, which can be used in conjunction with the *Person identifier flag* to identify these contacts. All other data elements on the *Service contact* record should be reported where these are available.
- For each unique patient identifier generated for a non-uniquely identifiable client reported in *Service contact details* records, a corresponding *Person details* record should be created. The various data elements normally included in *Person details* records should be reported here where these are available.

3.3. Options for how patient identifiers for non-uniquely identifiable clients can be generated

There are many ways to achieve this. Below are suggestions for two common tools—SPSS and Microsoft Access. Further advice in relation to other packages can be provided to jurisdictions should this be required.

Note that it does not matter if patient identifier values for non-uniquely identifiable clients clash with real identifiers since there is a flag that allows them to be differentiated.

The general method suggested below assumes that the data source for non-uniquely identifiable clients is contained in a file/table with a single row containing the demographic and contact details. It also assumes that you have already created the registered client data set and that these are still in a format manipulable by your software tool of choice.

The steps are:

3.3.1. SPSS

1. Add a unique identifier to each row if one does not exist. You can use the following idiom to create the values:

```
COMPUTE PersIdent = $CASENUM.
```

2. Add the data from the unregistered clients to the tables/files/spreadsheets containing the registered client data making sure that the Person identifier flag is set correctly:

```
SAVE OUTFILE <temporary path for unregistered data>.

ADD FILES=* /FILE=<path to your registered **person** details data>
/KEEP <space separated list of **person** details items>.

SAVE OUTFILE <path to your final **person** details data>.

GET FILE <temporary path for unregistered data>.

ADD FILES=* /FILE=<path to your registered **service contact**
details data> /KEEP <space separated list of **service contact**
details items>.

SAVE OUTFILE <path to your final **service contact** details data>.
```

3.3.2. Microsoft Access

1. Add a unique identifier to each row if one does not exist:

- Create a table which contains all of the required columns including 'Person Identifier'. Ensure that the 'Person Identifier' column is of type AutoNumber.
- Append the patient data to that table excluding 'Person Identifier' from the list of fields to append. Each row will be given a unique number.

2. Add the data from the unregistered clients to the tables/files/spreadsheets containing the registered client data making sure that the Person identifier flag is set correctly. Run two append queries:

- One to append the person details records from your unregistered client table to your final person details table (already containing your registered person details records).
- The other to append the service contact details records to the final service contact details table.

4. Appendix B: Rules and Virtual Elements

4.1. Rules

4.1.1. AdultAgeInYthOrCAUnitHigh

Class:

Anomaly

Priority:

High

Message:

Inappropriate Ages (**\$bad**) are over 34, **\$prop.perc**) for unit (TP **\$TP**)

Mark:

SERV

Description:

Age at Contact is over 34 years, but unit target population is Youth or Child and Adolescent

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop as TP,
       sum((Age >= 35) :: int) as bad,
       sd_div_safe(sum((Age >= 35) :: int), count(*), 3) as prop
from SERV
join ConAge using (State, RegId, OrgId, ClusId, SUIId)
where TargetPop in ('5', '1')
group by State,
         RegId,
         OrgId,
         ClusId,
         SUIId,
         TargetPop
having count(*) > 1000
       and sd_div_safe(sum((Age >= 35) :: int), count(*), 3) > 0.10
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)
- [TargetPop](#)

Virtual Elements:

- [ConAge](#)

4.1.2. AdultAgeInYthOrCAUnitLow

Class:

Anomaly

Priority:

Low

Message:

Inappropriate Ages (\$bad) are between 25 and 34, \$prop.perc) for unit (TP \$TP)

Mark:

SERV

Description:

Age at Contact is between 25 and 34 years, but unit target population is Youth or Child and Adolescent

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop as TP,
       sum((Age between 25 and 34) :: int) as bad,
       sd_div_safe(sum((Age between 25 and 34) :: int), count(*), 3) as prop
from SERV
join ConAge using (State, RegId, OrgId, ClusId, SUIId)
where TargetPop in ('5', '1')
group by State,
         RegId,
         OrgId,
         ClusId,
         SUIId,
         TargetPop
having count(*) > 1000
       and sd_div_safe(sum((Age between 25 and 34) :: int), count(*), 3) > 0.10
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)
- [TargetPop](#)

Virtual Elements:

- [ConAge](#)

4.1.3. BadDxPrincAd

Class:

Inconsistent

Priority:

Low

Message:

Principal Diagnosis (**\$DxPrinc**) and Age (**\$Age**) less than 15

Mark:

CON.DxPrinc

Description:

The following diagnosis codes should not apply to ages less than 15: 'F03 ', 'F01.0 ', 'F01.1 ', 'F01.2 ', 'F01.3 ', 'F01.8 ', 'F01.9 '

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       DxPrinc,
       Age
from CON
join ConAge using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId, RecordId)
where DxPrinc in ('F03 ', 'F01.0 ', 'F01.1 ', 'F01.2 ', 'F01.3 ', 'F01.8 ', 'F01.9 ')
and Age < 15
```

Data Elements:

- ClusId
- DxPrinc
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

Virtual Elements:

- ConAge

4.1.4. BadDxPrincF**Class:**

Inconsistent

Priority:

High

Message:

Principal Diagnosis (**\$DxPrinc**) and Sex (**\$Sex**) is not female

Mark:

CON.DxPrinc

Description:

The following diagnosis codes should only apply to females: 'F53.0', 'F53.1', 'F53.8', 'F53.9', 'F32.01', 'F32.11', 'F32.21', 'F32.31', 'F32.81', 'F32.91', 'O99.3', 'F52.5'

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       DxPrinc,
       Sex
from CON
join PER using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
where DxPrinc in ('F53.0', 'F53.1', 'F53.8', 'F53.9', 'F32.01', 'F32.11',
                  'F32.21', 'F32.31', 'F32.81', 'F32.91', 'O99.3', 'F52.5')
       and Sex is not null
       and Sex != '2'
       and Sex != '3'
```

Data Elements:

- ClusId
- DxPrinc
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- Sex
- State

4.1.5. BadDxPrincLowAge**Class:**

Inconsistent

Priority:

Low

Message:

Principal Diagnosis (\$DxPrinc) and Age (\$Age) less than 1

Mark:

CON.DxPrinc

Description:

The following diagnosis codes should not apply to ages less than 1: 'F80.0', 'F80.1', 'F80.2', 'F80.3', 'F80.8', 'F80.9', 'F81.0', 'F81.1', 'F81.2', 'F81.3', 'F81.8', 'F81.9', 'F82', 'F83', 'F88', 'F89'

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       DxPrinc,
       Age
from CON
join ConAge using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId, RecordId)
where DxPrinc in ('F80.0 ', 'F80.1 ', 'F80.2 ', 'F80.3 ', 'F80.8 ', 'F80.9 ', 'F81.0 ',
                  'F81.1 ', 'F81.2 ', 'F81.3 ', 'F81.8 ', 'F81.9 ', 'F82   ', 'F83   ', 'F88   ',
                  'F89   ')
and Age < 1
```

Data Elements:

- ClusId
- DxPrinc
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

Virtual Elements:

- ConAge

4.1.6. BadDxPrincM

Class:

Inconsistent

Priority:

High

Message:

Principal Diagnosis (**\$DxPrinc**) and Sex (**\$Sex**) is not male

Mark:

CON.DxPrinc

Description:

The following diagnosis codes should only apply to males: 'F52.4 '

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       DxPrinc,
       Sex
from CON
join PER using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
where DxPrinc in ('F52.4 ')
       and Sex is not null
       and Sex != '1'
       and Sex != '3'
```

Data Elements:

- ClusId
- DxPrinc
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- Sex
- State

4.1.7. BadDxPrincPpm

Class:

Inconsistent

Priority:

Low

Message:

Principal Diagnosis (\$DxPrinc) and Age (\$Age) not between 10 and 60

Mark:

CON.DxPrinc

Description:

The following diagnosis codes should only apply to ages 10-60: 'F53 ' , 'F53.0 ' , 'F53.1 ' , 'F53.8 ' , 'F53.9 ' , 'F32.01' , 'F32.11' , 'F32.21' , 'F32.31' , 'F32.81' , 'F32.91' , 'O99.3 '

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       DxPrinc,
       Age
from CON
join ConAge using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId, RecordId)
where DxPrinc in ('F53 ', 'F53.0 ', 'F53.1 ', 'F53.8 ', 'F53.9 ', 'F32.01',
                  'F32.11', 'F32.21', 'F32.31', 'F32.81', 'F32.91', '099.3 ')
and Age not between 10 and 60
```

Data Elements:

- ClusId
- DxPrinc
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

Virtual Elements:

- ConAge

4.1.8. BadHrLegalSt12Counts

Class:

Anomaly

Priority:

Medium

Message:

LegalSt '1' count (\$Val1Count) exceeds '2' count (\$Val2Count)

Mark:

HR

Description:

Mental Health Legal Status (LegalSt) '1' (Involuntary patient) count exceeds '2' (Voluntary patient) count

SQL:

```
select State,
       Val1.Count as Val1Count,
       Val2.Count as Val2Count
from HrLegalStInvolCount as Val1
join HrLegalStVolCount as Val2 using (State)
where Val1.Count > Val2.Count
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrLegalStInvolCount](#)
- [HrLegalStVolCount](#)

4.1.9. BadHrPersIdFlag21Counts

Class:

Anomaly

Priority:

High

Message:

PersIdFlag '2' count ([\\$Val2Count](#)) exceeds '1' count ([\\$Val1Count](#))

Mark:

HR

Description:

Person Identifier Flag (PersIdFlag) '2' (No, Patient identifier is for a non-uniquely identifiable...) count exceeds '1' (Yes, Patient identifier is for a uniquely identifiable...) count

SQL:

```
select State,
       Val2.Count as Val2Count,
       Val1.Count as Val1Count
  from HrPersIdFlagDummyCount as Val2
 join HrPersIdFlagRealCount as Val1 using (State)
 where Val2.Count > Val1.Count
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrPersIdFlagDummyCount](#)
- [HrPersIdFlagRealCount](#)

4.1.10. BadSA2Prop

Class:

Anomaly

Priority:

Low

Message:

Greater than 5% ([\\$prop.perc](#)) of SA2s are invalid ([\\$BadCount](#) in total)

Mark:

HR.State

Description:

Greater than 5% of SA2s (ResArea) are invalid

SQL:

```
select State,
       count(*) as AllCount,
       bad.BadCount,
       sd_div_safe(bad.BadCount, count(*), 3) as prop
from CON
cross join (
  select count(*) as BadCount
  from error
  join rule
  on (rule.id = error.rule)
  where rule.name = 'Domain'
  and error.field = 'ResArea'
) as bad
group by State,
       bad.BadCount
having sd_div_safe(bad.BadCount, count(*), 3) > 0.05
```

Data Elements:

- [ResArea](#)
- [State](#)

4.1.11. ClusClusNameMissing

Class:

Missing

Priority:

High

Message:

Missing data - ClusName \$ClusName.q

Mark:

CLUS.ClusName

Description:

Missing data - Service Unit Cluster Name (ClusName)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       ClusName
from CLUS
where ClusName is null
```

Data Elements:

- [ClusId](#)
- [ClusName](#)
- [OrgId](#)
- [RegId](#)
- [State](#)

4.1.12. ConContDtMissing

Class:

Missing

Priority:

High

Message:

Missing data - ContDt `$ContDt.q`

Mark:

CON.ContDt

Description:

Missing data - Service Contact Date (ContDt)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDt
from CON
where ContDt is null
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.13. ConContDurMissing

Class:

Missing

Priority:

High

Message:

Missing data - ContDur `$ContDur.q`

Mark:

CON.ContDur

Description:

Missing data - Mental Health Service Contact Duration (ContDur)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDur
from CON
where ContDur is null
```

Data Elements:

- [ClusId](#)
- [ContDur](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.14. ConContDurZero

Class:

Anomaly

Priority:

High

Message:

Zero reported for ContDur

Mark:

CON.ContDur

Description:

Zero reported for Mental Health Service Contact Duration

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDur as value
from CON
where ContDur = 0
```

Data Elements:

- [ClusId](#)
- [ContDur](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.15. ConContParticMissing**Class:**

Missing

Priority:

High

Message:

Missing data - ContPartic \$ContPartic.q

Mark:

CON.ContPartic

Description:

Missing data - Mental Health Service Contact-Patient/Client Participation Indicator (ContPartic)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContPartic
from CON
where ContPartic is null
```

Data Elements:

- [ClusId](#)
- [ContPartic](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.16. ConContSessTypeMissing

Class:

Missing

Priority:

High

Message:

Missing data - ContSessType `$ContSessType.q`

Mark:

CON.ContSessType

Description:

Missing data - Mental Health Service Contact-Session Type (ContSessType)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContSessType
from CON
where ContSessType is null
```

Data Elements:

- [ClusId](#)
- [ContSessType](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.17. ConInvolAndUnreg

Class:

Inconsistent

Priority:

High

Message:

PersIdFlag is 2 (dummy) and LegalSt is 1 (involuntary)

Mark:

CON.LegalSt

Description:

CON record with a dummy PersId (PersIdFlag 2) has an Involuntary legal status (LegalSt 1). Clients with an Involuntary legal status should be registered. PersIdFlag 1: No, Patient identifier is for a non-uniquely identifiable... LegalSt 1: Involuntary patient

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       LegalSt
from CON
where PersIdFlag = '2'
and LegalSt = '1'
```

Data Elements:

- [ClusId](#)
- [LegalSt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.18. ConLegalStMissing**Class:**

Missing

Priority:

High

Message:

Missing data - LegalSt \$LegalSt.q

Mark:

CON.LegalSt

Description:

Missing data - Mental Health Legal Status (LegalSt)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       LegalSt
from CON
where LegalSt is null
```

Data Elements:

- [ClusId](#)
- [LegalSt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.19. ConMaritalStMiscoded

Class:

Invalid

Priority:

High

Message:

MaritalSt contains spaces instead of appropriate value

Mark:

CON.MaritalSt

Description:

MaritalSt should not contain spaces. To indicate a missing value, the appropriate numeral should be given here

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       CON.MaritalSt
from CON
where CON.MaritalSt is null
```

Data Elements:

- [ClusId](#)
- [MaritalSt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.20. ConResAreaMiscoded**Class:**

Invalid

Priority:

High

Message:

ResArea contains spaces instead of appropriate value

Mark:

CON.ResArea

Description:

ResArea should not contain spaces. To indicate a missing value, the appropriate numeral should be given here

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       CON.ResArea
from CON
where CON.ResArea is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [ResArea](#)
- [SUIId](#)
- [State](#)

4.1.21. ContDtBeforeDoB

Class:

Inconsistent

Priority:

High

Message:

ContDt (`$ContDt.dmy`) is before DoB (`$DoB.dmy`)

Mark:

CON.ContDt

Description:

ContDt is before DoB

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDt,
       DoB
from CON
join PER using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
where DoB != '9999-09-09'
and ContDt < DoB
AND DoBFlag IN ('1', '2')
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [DoB](#)
- [DoBFlag](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.22. ContDtOutsideCollection

Class:

Anomaly

Priority:

Low

Message:

Contact Date outside collection period (\$ContDt)

Mark:

CON.ContDt

Description:

Contact Date is outside the collection period of the file

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDt
from CON
JOIN HR using (State)
where ContDt < HR.repstart
      OR ContDt > HR.repend
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.23. ContParticChange

Class:

Historical

Priority:

Low

Message:

Client participation has changed by \$PercChange percent.

Mark:

HR.State

Description:

Variation over 15 percent in client participation

SQL:

```
select State,
       round(100::float * abs(New.prop - Old.prop)) as PercChange
from HrContParticProp as New
join hist.HrContParticProp as Old using(State)
where abs(New.prop - Old.prop) > 0.15;
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrContParticProp](#)

4.1.24. HighAge

Class:

Anomaly

Priority:

Low

Message:

Age is greater than 124 years

Mark:

CON

Description:

Age at Contact is greater than 124 years

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       Age
from ConAge
where Age > 124
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

Virtual Elements:

- [ConAge](#)

4.1.25. HighBusyConDayCount

Class:

Anomaly

Priority:

High

Message:

Person has 16 or more (**\$Count**) contacts on one day (**\$ContDt.dmy**) totalling more than 600 minutes
(**\$TotalContDur** mins)

Mark:

PER

Description:

Person has 16 or more contact records within a service unit on a single day totalling more than 600 minutes

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       ContDt,
       TotalContDur,
       Count
from PER
join (
    select State,
           RegId,
           OrgId,
           ClusId,
           SUIId,
           PersIdFlag,
           PersId,
           ContDt,
           sum(ContDur) as TotalContDur,
           count(*) as Count
    from CON
    group by State,
             RegId,
             OrgId,
             ClusId,
             SUIId,
             PersIdFlag,
             PersId,
             ContDt
    having count(*) > 15
           AND SUM(ContDur) >= 600
) tmpperconcount using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [ContDur](#)

- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.26. HighConCount

Class:

Anomaly

Priority:

Low

Message:

Person has over 250 (\$Count) contacts

Mark:

PER

Description:

Person has over 250 contact records within a service unit

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       count(*) as Count
from CON
group by State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId
having count(*) > 250
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.27. HighConDayCount

Class:

Anomaly

Priority:

Low

Message:

Person has 16 or more contact records within a service unit on \$days days, each day totalling less than 600 minutes

Mark:

PER

Description:

Person has 16 or more contact records within a service unit on one or more days, each day totalling less than 600 minutes

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       COUNT(*) as days
from PER
join (
    select State,
           RegId,
           OrgId,
           ClusId,
           SUIId,
           PersIdFlag,
           PersId,
           ContDt,
           sum(ContDur) as TotalContDur,
           count(*) as Count
    from CON
    group by State,
             RegId,
             OrgId,
             ClusId,
             SUIId,
             PersIdFlag,
             PersId,
             ContDt
    having count(*) > 15
       AND SUM(ContDur) <= 599
) tmppperconcount using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
GROUP BY State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [ContDur](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.28. HighConDtDayProp**Class:**

Anomaly

Priority:

Low

Message:

One day ([\\$ContDtDay](#)) has over 4% of all contacts

Mark:

HR.State

Description:

One day has greater than 4% of all contacts (\$ConPercent.perc), indicating it may be a default date or suffer from incomplete reporting

SQL:

```
select State,
       ContDtDay,
       sd_div_safe(ConCount, StConCountTotal, 3) as ConPercent
from StConCount,
(
  select State,
         extract(day FROM ContDt) AS ContDtDay,
         count(*) as ConCount
  from CON
  group by State,
           ContDtDay
) as tmpservcontotals
where sd_div_safe(ConCount, StConCountTotal, 3) > 0.04
```

Data Elements:

- [ContDt](#)
- [State](#)

4.1.29. HighConDtMonthProp**Class:**

Anomaly

Priority:

Low

Message:

One month (`$ContDtMonth`) has over 10% of all contacts (`$ConPercent.perc`)

Mark:

HR.State

Description:

One month has over 10% of all contacts, indicating it may be a default date or suffer from incomplete reporting

SQL:

```
select State,
       ContDtMonth,
       sd_div_safe(ContDtCountSum, StConCountTotal, 3) as ConPercent
from StConCount,
(
  select State,
         ContDtMonth,
         sum(ContDtCount) as ContDtCountSum
  from ServConCountByMonth
  group by State,
         ContDtMonth
) as tmpservcontotals
where sd_div_safe(ContDtCountSum, StConCountTotal, 3) > 0.10
```

Data Elements:

- [State](#)

4.1.30. HighConDtYearProp

Class:

Anomaly

Priority:

Low

Message:

One half-financial-year (`$ContDtYear`) has over 60% of all contacts (`$ConPercent.perc`)

Mark:

HR.State

Description:

One year (ie. one half of the financial year) has greater than 60% of all contacts, indicating it may include a default date or suffer from incomplete reporting.

SQL:

```
select State,
       ContDtYear,
       sd_div_safe(ContDtCountSum, StConCountTotal, 3) as ConPercent
from StConCount,
(
  select State,
         ContDtYear,
         sum(ContDtCount) as ContDtCountSum
  from ServConCountByMonth
  group by State,
         ContDtYear
) as tmpservcontotals
where sd_div_safe(ContDtCountSum, StConCountTotal, 3) > 0.60
```

Data Elements:

- [State](#)

4.1.31. HighContDur

Class:

Anomaly

Priority:

Low

Message:

Duration is greater than 480 minutes ([\\$ContDur](#))

Mark:

CON.ContDur

Description:

Contact Duration is greater than 480 minutes

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDur
from CON
where ContDur > 480
```

Data Elements:

- [ClusId](#)
- [ContDur](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)

- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.32. HighEstDoBFlagProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 15% ([\\$prop.perc](#)) of Persons have an Estimated Date of Birth

Mark:

HR.State

Description:

Greater than 15% of Persons have an Estimated Date of Birth

SQL:

```
select State,
        prop
  from HrDoBFlagPropRegistered
 where prop > 0.15
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrDoBFlagPropRegistered](#)

4.1.33. HighExtCoBProp

Class:

Anomaly

Priority:

High

Message:

Greater than 50% ([\\$prop.perc](#)) of Persons have a CoB other than Australia

Mark:

HR.State

Description:

Greater than 50% of Persons have a CoB other than Australia

SQL:

```
select State,
        prop
  from HrCoBPropNotAus
 where prop > 0.5
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrCoBPropNotAus](#)

4.1.34. HighExtSA2Prop

Class:

Anomaly

Priority:

Low

Message:

Greater than 5% ([\\$prop.perc](#)) of Contacts have a ResArea out of state

Mark:

HR.State

Description:

Greater than 5% of Contacts a ResArea out of state. (This check does not apply to ACT)

SQL:

```
select State,
        prop
  from HrResAreaProp
 where State != '8'
        and prop > 0.05
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrResAreaProp](#)

4.1.35. HighIndigNonAustProp

Class:

Anomaly

Priority:

High

Message:

Proportion of IndigSt not born in Australia is greater than 5% ([\\$prop.perc](#))

Mark:

ORG

Description:

Proportion of Indigenous Australians not born in Australia is greater than 5%

SQL:

```
select State,
        RegId,
        OrgId,
        prop
  from OrgIndigStPropNonAus
 where prop > 0.05
```

Data Elements:

- [OrgId](#)

- [RegId](#)
- [State](#)

Virtual Elements:

- [OrgIndigStPropNonAus](#)

4.1.36. HighMissingLegalStProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 5% ([\\$prop.perc](#)) of Contacts have Missing as a legal status

Mark:

HR.State

Description:

Greater than 5% of Contacts have Missing legal status

SQL:

```
select State,
        prop
  from HrLegalStPropRegistered
 where prop > 0.05
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrLegalStPropRegistered](#)

4.1.37. HighPersIdFlagProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 10% ([\\$prop.perc](#)) of Persons have a dummy PersId

Mark:

HR.State

Description:

Greater than 10% of Persons have a dummy PersId

SQL:

```
select State,
        prop
  from HrPersIdFlagProp
 where prop > 0.1
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrPersIdFlagProp](#)

4.1.38. HighSuppCoBProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 15% ([\\$prop.perc](#)) of Persons have a CoB with Supplementary code

Mark:

HR.State

Description:

Greater than 15% of Persons have a CoB with Supplementary code

SQL:

```
select State,
        prop
  from HrCoBPropSuppRegistered
 where prop > 0.15
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrCoBPropSuppRegistered](#)

4.1.39. HighUnknownContParticProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 15% of contacts ([\\$PercChange](#) %) have Unknown ContPartic

Mark:

ORG

Description:

Greater than 15% of contacts have Unknown client participation status

SQL:

```
select State,
        RegId,
        OrgId,
        round(100::float * abs(prop)) as PercChange
  from OrgUnknownContParticProp
 where abs(prop) > 0.15;
```

Data Elements:

- [OrgId](#)

- [RegId](#)
- [State](#)

Virtual Elements:

- [OrgUnknownContParticProp](#)

4.1.40. HighUnknownContSessTypeProp

Class:

Anomaly

Priority:

Low

Message:

Greater than 15% of contacts ([\\$PercChange](#) %) have Unknown ContSessType

Mark:

ORG

Description:

Greater than 15% of contacts have Unknown session type status

SQL:

```
select State,
       RegId,
       OrgId,
       round(100::float * abs(prop)) as PercChange
from OrgUnknownContSessTypeProp
where abs(prop) > 0.15;
```

Data Elements:

- [OrgId](#)
- [RegId](#)
- [State](#)

Virtual Elements:

- [OrgUnknownContSessTypeProp](#)

4.1.41. HrGenDtMissing

Class:

Missing

Priority:

High

Message:

Missing data - GenDt [\\$GenDt.q](#)

Mark:

HR.GenDt

Description:

Missing data - Data File Generation Date (GenDt)

SQL:

```
select State,
        GenDt
  from HR
 where GenDt is null
```

Data Elements:

- [GenDt](#)
- [State](#)

4.1.42. LowAge

Class:

Anomaly

Priority:

Low

Message:

Age is less than 1 years ([\\$Age](#))

Mark:

CON

Description:

Age at Contact is less than 1 years

SQL:

```
select State,
        RegId,
        OrgId,
        ClusId,
        SUIId,
        PersIdFlag,
        PersId,
        RecordId,
        Age
  from ConAge
 where Age < 1
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RecordId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

Virtual Elements:

- [ConAge](#)

4.1.43. LowAgeInOldUnitHigh

Class:

Anomaly

Priority:

High

Message:

Inappropriate Ages (\$bad) are below 25, \$prop.perc) for unit (TP \$TP)

Mark:

SERV

Description:

Age at Contact is below 25 years, but unit target population is Older person

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop as TP,
       sum((Age <= 24) :: int) as bad,
       sd_div_safe(sum((Age <= 24) :: int), count(*), 3) as prop
from SERV
join ConAge using (State, RegId, OrgId, ClusId, SUIId)
where TargetPop in ('2')
group by State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop
having count(*) > 1000
and sd_div_safe(sum((Age <= 24) :: int), count(*), 3) > 0.10
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)
- [TargetPop](#)

Virtual Elements:

- [ConAge](#)

4.1.44. LowAgeInOldUnitLow

Class:

Anomaly

Priority:

Low

Message:

Inappropriate Ages (`$bad`) are between 25 and 34, `$prop.perc`) for unit (TP `$TP`)

Mark:

SERV

Description:

Age at Contact is between 25 and 34 years, but unit target population is Older person

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop as TP,
       sum((Age between 25 and 34) :: int) as bad,
       sd_div_safe(sum((Age between 25 and 34) :: int), count(*), 3) as prop
from SERV
join ConAge using (State, RegId, OrgId, ClusId, SUIId)
where TargetPop in ('2')
group by State,
         RegId,
         OrgId,
         ClusId,
         SUIId,
         TargetPop
having count(*) > 1000
       and sd_div_safe(sum((Age between 25 and 34) :: int), count(*), 3) > 0.10
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)
- [TargetPop](#)

Virtual Elements:

- [ConAge](#)

4.1.45. LowAgeMarriageHigh**Class:**

Anomaly

Priority:

Medium

Message:

Age is less than 13 years and MaritalSt is `$MaritalSt`

Mark:

CON.MaritalSt

Description:

Age at Contact is less than 13 years and Marital Status is not 1 or 6 (Never married or Not stated)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       Age,
       MaritalSt
from CON
join ConAge using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId, RecordId)
where Age < 13
and MaritalSt NOT IN ('1','6')
```

Data Elements:

- ClusId
- MaritalSt
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

Virtual Elements:

- ConAge

4.1.46. LowAgeMarriageLow

Class:

Anomaly

Priority:

Low

Message:

Age is 13 to 15 years and MaritalSt is \$MaritalSt

Mark:

CON.MaritalSt

Description:

Age at Contact is 13 to 15 years and Marital Status is not 1 or 6 (Never married or Not stated)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       Age,
       MaritalSt
from CON
join ConAge using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId, RecordId)
where Age < 16
       and Age >= 13
       and MaritalSt NOT IN ('1','6')
```

Data Elements:

- ClusId
- MaritalSt
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

Virtual Elements:

- ConAge

4.1.47. LowContDur

Class:

Anomaly

Priority:

Low

Message:

Duration is less than 5 minutes but greater than 0 minutes (\$ContDur)

Mark:

CON.ContDur

Description:

Contact Duration is less than 5 minutes but greater than 0 minutes

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       ContDur
from CON
where ContDur > 0
      and ContDur < 5
```

Data Elements:

- ClusId
- ContDur
- OrgId
- PersId
- PersIdFlag
- RecordId
- RegId
- SUIId
- State

4.1.48. MedConDayCount

Class:

Anomaly

Priority:

Low

Message:

Person has over 10 but less than 16 (**\$Count**) contacts on one day (**\$ContDt.dmy**); TotalContDur
\$TotalContDur mins

Mark:

PER

Description:

Person has over 10 but less than 16 contact records within a service unit on a single day

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       ContDt,
       TotalContDur,
       Count
from PER
join (
    select State,
           RegId,
           OrgId,
           ClusId,
           SUIId,
           PersIdFlag,
           PersId,
           ContDt,
           sum(ContDur) as TotalContDur,
           count(*) as Count
    from CON
    group by State,
             RegId,
             OrgId,
             ClusId,
             SUIId,
             PersIdFlag,
             PersId,
             ContDt
    having count(*) > 10
           and count(*) < 16
) tmpperconcount using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [ContDt](#)
- [ContDur](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.49. OrgCAInSklOnly

Class:

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: CA)

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file (TargetPop: CA)

4.1.50. OrgCANotInSkl**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: CA)

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data (TargetPop: CA)

4.1.51. OrgForInSklOnly**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: For)

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file (TargetPop: For)

4.1.52. OrgForNotInSkl**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: For)

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data (TargetPop: For)

4.1.53. OrgGenInSklOnly**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Gen)

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file (TargetPop: Gen)

4.1.54. OrgGenNotInSkl**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Gen)

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data (TargetPop: Gen)

4.1.55. OrgInSklOnly**Class:**

Skeleton

Priority:

High

Message:

Org \$name expected from SKL is missing

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file

4.1.56. OrgInvolGrowthVaries**Class:**

Historical

Priority:

Medium

Message:

Growth variation over 5% (\$Growth.perc) in ratio of Involuntary legal status

Mark:

ORG

Description:

Organisation-wide ratio of registered contacts with involuntary legal status has increased by more than 5 per cent from the previous year.

SQL:

```
select State,
       RegId,
       OrgId,
       round(Growth, 3) as Growth
from OrgLegalStInvolRatioGrowth
where (Growth) > 0.5
```

Data Elements:

- OrgId
- RegId
- State

Virtual Elements:

- OrgLegalStInvolRatioGrowth

4.1.57. OrgNotInSkl

Class:

Skeleton

Priority:

High

Message:

Org \$name not in SKL data

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data

4.1.58. OrgOldInSklOnly

Class:

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Old)

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file (TargetPop: Old)

4.1.59. OrgOldNotInSkl

Class:

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Old)

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data (TargetPop: Old)

4.1.60. OrgOrgNameMissing**Class:**

Missing

Priority:

High

Message:

Missing data - OrgName \$OrgName.q

Mark:

ORG.OrgName

Description:

Missing data - Organisation Name (OrgName)

SQL:

```
select State,
        RegId,
        OrgId,
        OrgName
from ORG
where OrgName is null
```

Data Elements:

- [OrgId](#)
- [OrgName](#)
- [RegId](#)
- [State](#)

4.1.61. OrgYthInSkIOnly**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Yth)

Description:

Organisation appears in skeleton reference data only - A Organisation with matching Ids is expected based on the SKL data but is not present in this file (TargetPop: Yth)

4.1.62. OrgYthNotInSkI**Class:**

Skeleton

Priority:

High

Message:

Org \$name not in SKL data (TargetPop: Yth)

Description:

Organisation not in skeleton reference data - A matching Ambulatory Organisation was not found in the skeleton data (TargetPop: Yth)

4.1.63. PerCoBDiffers

Class:

Inconsistent

Priority:

High

Message:

Person has \$attr_count values for CoB (\$attr_vals)

Mark:

PER.CoB

Description:

Person has multiple values for CoB (Country of Birth) across one organisation

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       attr_count,
       attr_vals
from PER
join (
    select State,
           RegId,
           OrgId,
           PersIdFlag,
           PersId,
           count(DISTINCT CoB) as attr_count,
           string_agg(DISTINCT CoB::TEXT, ',') as attr_vals
    from PER
    group by State,
             RegId,
             OrgId,
             PersIdFlag,
             PersId
    having count(DISTINCT CoB) > 1
) as foo using (State, RegId, OrgId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [CoB](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)

- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.64. PerCoBMiscoded

Class:

Invalid

Priority:

High

Message:

CoB contains spaces instead of appropriate value

Mark:

PER.CoB

Description:

CoB should not contain spaces. To indicate a missing value, the appropriate numeral should be given here

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       PER.CoB
from PER
where PER.CoB is null
```

Data Elements:

- [ClusId](#)
- [CoB](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.65. PerDoBCount

SQL:

```
select sum(DoBCount) as PerDoBCountTotal,  
       avg(DoBCount) as PerDoBCountAvg  
from (  
    select count(*) as DoBCount  
    from PER  
    where DoB <> '9999-09-09'  
    group by DoB  
    ) as tmpperdobcount
```

Data Elements:

- DoB

4.1.66. PerDoBDiffers

Class:

Inconsistent

Priority:

High

Message:

Person has \$attr_count values for DoB (\$attr_vals)

Mark:

PER.DoB

Description:

Person has multiple values for DoB (Date of Birth) across one organisation

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       attr_count,
       attr_vals
from PER
join (
    select State,
           RegId,
           OrgId,
           PersIdFlag,
           PersId,
           count(DISTINCT DoB) as attr_count,
           string_agg(DISTINCT DoB::TEXT, ',') as attr_vals
    from PER
    group by State,
             RegId,
             OrgId,
             PersIdFlag,
             PersId
    having count(DISTINCT DoB) > 1
) as foo using (State, RegId, OrgId, PersIdFlag, PersId)
```

Data Elements:

- ClusId
- DoB
- OrgId
- PersId
- PersIdFlag
- RegId
- SUIId
- State

4.1.67. PerDoBFlagAndDoB

Class:

Inconsistent

Priority:

High

Message:

DoBFlag is 8 and DoB is not 09099999 (\$BadDoB.ddmmyyyy)

Mark:

PER.DoB

Description:

If date of birth flag (DoBFlag) is '8' (dummy date), date of birth (DoB) must be '09099999'. DoBFlag 8: Date of birth is a "dummy" date (ie, 09099999)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       DoBFlag,
       DoB as BadDoB
from PER
where DoBFlag = '8'
and DoB != '9999-09-09'
```

Data Elements:

- ClusId
- DoB
- DoBFlag
- OrgId
- PersId
- PersIdFlag
- RegId
- SUIId
- State

4.1.68. PerDoBFlagDiffers

Class:

Inconsistent

Priority:

High

Message:

Person has \$attr_count values for DoBFlag (\$attr_vals)

Mark:

PER.DoBFlag

Description:

Person has multiple values for DoBFlag (Estimated Date of Birth Flag) across one organisation

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       attr_count,
       attr_vals
from PER
join (
    select State,
           RegId,
           OrgId,
           PersIdFlag,
           PersId,
           count(DISTINCT DoBFlag) as attr_count,
           string_agg(DISTINCT DoBFlag::TEXT, ',') as attr_vals
    from PER
    group by State,
             RegId,
             OrgId,
             PersIdFlag,
             PersId
    having count(DISTINCT DoBFlag) > 1
) as foo using (State, RegId, OrgId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [DoBFlag](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.69. PerDoBFlagMissing

Class:

Missing

Priority:

High

Message:

Missing data - DoBFlag \$DoBFlag.q

Mark:

PER.DoBFlag

Description:

Missing data - Estimated Date of Birth Flag (DoBFlag)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       DoBFlag
from PER
where DoBFlag is null
```

Data Elements:

- [ClusId](#)
- [DoBFlag](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.70. PerDoBMissing

Class:

Missing

Priority:

High

Message:

Missing data - DoB \$DoB.q

Mark:

PER.DoB

Description:

Missing data - Date of Birth (DoB)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       DoB
from PER
where DoB is null
```

Data Elements:

- [ClusId](#)
- [DoB](#)

- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.71. PerIndigStDiffers

Class:

Inconsistent

Priority:

High

Message:

Person has `$attr_count` values for IndigSt (`$attr_vals`)

Mark:

PER.IndigSt

Description:

Person has multiple values for IndigSt (Indigenous Status) across one organisation

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       attr_count,
       attr_vals
from PER
join (
    select State,
           RegId,
           OrgId,
           PersIdFlag,
           PersId,
           count(DISTINCT IndigSt) as attr_count,
           string_agg(DISTINCT IndigSt::TEXT, ',') as attr_vals
    from PER
    group by State,
             RegId,
             OrgId,
             PersIdFlag,
             PersId
    having count(DISTINCT IndigSt) > 1
) as foo using (State, RegId, OrgId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [IndigSt](#)
- [OrgId](#)

- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.72. PerIndigStMissing

Class:

Missing

Priority:

High

Message:

Missing data - IndigSt `$IndigSt.q`

Mark:

PER.IndigSt

Description:

Missing data - Indigenous Status (IndigSt)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       IndigSt
from PER
where IndigSt is null
```

Data Elements:

- [ClusId](#)
- [IndigSt](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.73. PerSexDiffers

Class:

Inconsistent

Priority:

High

Message:

Person has `$attr_count` values for Sex (`$attr_vals`)

Mark:

PER.Sex

Description:

Person has multiple values for Sex (Sex) across one organisation

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       attr_count,
       attr_vals
from PER
join (
    select State,
           RegId,
           OrgId,
           PersIdFlag,
           PersId,
           count(DISTINCT Sex) as attr_count,
           string_agg(DISTINCT Sex::TEXT, ',') as attr_vals
    from PER
    group by State,
             RegId,
             OrgId,
             PersIdFlag,
             PersId
    having count(DISTINCT Sex) > 1
) as foo using (State, RegId, OrgId, PersIdFlag, PersId)
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [Sex](#)
- [State](#)

4.1.74. PerSexGenderMissing**Class:**

Missing

Priority:

High

Message:

Missing data - at least one of Sex `$Sex.q` or Gender `$Gender.q` required

Mark:

PER

Description:

Missing data - at least one of Sex (Sex) or Gender (Gender) required

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       Sex,
       Gender
from PER
where (coalesce(cast(Sex in ('1','2','3','9') as integer),0) + coalesce(cast(Gender
in ('1','2','3','4','5','9') as integer),0) + 0) = 0
```

Data Elements:

- [ClusId](#)
- [Gender](#)
- [OrgId](#)
- [PersId](#)
- [PersIdFlag](#)
- [RegId](#)
- [SUIId](#)
- [Sex](#)
- [State](#)

4.1.75. RegInSkIOnly

Class:

Skeleton

Priority:

High

Message:

Reg `$name` expected from SKL is missing

Description:

Region appears in skeleton reference data only - A Region with matching Ids is expected based on the SKL data but is not present in this file

4.1.76. RegIndigStGrowthVaries

Class:

Historical

Priority:

Medium

Message:

Growth variation over 20% in IndigSt (\$PercChange %)

Mark:

REG

Description:

Proportion of Indigenous Australians has changed by more than 20% from the previous year

SQL:

```
select State,
       RegId,
       round(100::float * abs(New.prop - Old.prop)) as PercChange
from RegIndigStPropNonAus as New
join hist.RegIndigStPropNonAus as Old using(State, RegId)
where abs(New.prop - Old.prop) > 0.20;
```

Data Elements:

- [RegId](#)
- [State](#)

Virtual Elements:

- [RegIndigStPropNonAus](#)

4.1.77. RegNotInSkI

Class:

Skeleton

Priority:

High

Message:

Reg \$name not in SKL data

Description:

Region not in skeleton reference data - A matching Ambulatory Region was not found in the skeleton data

4.1.78. RegRegNameMissing

Class:

Missing

Priority:

High

Message:

Missing data - RegName \$RegName.q

Mark:

REG.RegName

Description:

Missing data - Region Name (RegName)

SQL:

```
select State,
        RegId,
        RegName
from REG
where RegName is null
```

Data Elements:

- [RegId](#)
- [RegName](#)
- [State](#)

4.1.79. SectorPrivate

Class:

Invalid

Priority:

High

Message:

Sector is \$Sector.qt (Private)

Mark:

SERV.Sector

Description:

Service Unit Sector must be 1 (Public)

SQL:

```
select State,
        RegId,
        OrgId,
        ClusId,
        SUIId,
        Sector
from SERV
where Sector = '2'
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [Sector](#)
- [State](#)

4.1.80. ServClosed

Class:

Historical

Priority:

Medium

Message:

Serv closed, historical `$hist_name.qt` (SUIId: `$hist_SUIId`) no longer exists

Mark:

HR.State

Description:

Service Unit Closed - A historical Service Unit was not found in current data

SQL:

```
select State,
       hist_entity.State as hist_State,
       hist_entity.RegId as hist_RegId,
       hist_entity.OrgId as hist_OrgId,
       hist_entity.ClusId as hist_ClusId,
       hist_entity.SUIId as hist_SUIId,
       hist_entity.SUName as hist_name
from hist.SERV as hist_entity
left join main.SERV using (State, RegId, OrgId, ClusId, SUIId)
where SERV.SUIId is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [SUName](#)
- [State](#)

4.1.81. ServConCount**SQL:**

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       sum(ContDtCount) as ServConCountTotal
from ServConCountByMonth
group by State,
       RegId,
       OrgId,
       ClusId,
       SUIId
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)

4.1.82. ServConCountByMonth

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       extract(month FROM ContDt) AS ContDtMonth,
       extract(year FROM ContDt) AS ContDtYear,
       count(*) as ContDtCount
from SERV
join CON using (State, RegId, OrgId, ClusId, SUIId)
group by State,
         RegId,
         OrgId,
         ClusId,
         SUIId,
         ContDtMonth,
         ContDtYear
```

Data Elements:

- ClusId
- ContDt
- OrgId
- RegId
- SUIId
- State

4.1.83. ServOpened

Class:

Historical

Priority:

Medium

Message:

Serv opened, \$SUName.qt (SUIId: \$SUIId) not in historical data

Mark:

SERV

Description:

Service Unit Opened - A matching Service Unit was not found in the historical data

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       SERV.SUName
from main.SERV
left join hist.SERV as hist_entity using (State, RegId, OrgId, ClusId, SUIId)
where hist_entity.SUIId is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [SUName](#)
- [State](#)

4.1.84. ServRenamed

Class:

Historical

Priority:

Medium

Message:

Serv renamed from `$hist_name.qt` to `$SUName.qt`

Mark:

SERV.SUName

Description:

Service Unit Renamed - Service Unit Name differs between historical and current data

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       SERV.SUName,
       hist_entity.SUName as hist_name
from SERV
join hist.SERV as hist_entity using(State, RegId, OrgId, ClusId, SUIId)
where not sloppy_match(SERV.SUName, hist_entity.SUName)
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)

- [SUName](#)
- [State](#)

4.1.85. ServSUNameMissing

Class:

Missing

Priority:

High

Message:

Missing data - SUName \$SUName.q

Mark:

SERV.SUName

Description:

Missing data - Ambulatory Service Unit Name (SUName)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       SUName
from SERV
where SUName is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [SUName](#)
- [State](#)

4.1.86. ServSectorMissing

Class:

Missing

Priority:

High

Message:

Missing data - Sector \$Sector.q

Mark:

SERV.Sector

Description:

Missing data - Sector (Sector)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       Sector
from SERV
where Sector is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [Sector](#)
- [State](#)

4.1.87. ServTargetPopChanged

Class:

Historical

Priority:

High

Message:

Service Unit TargetPop changed from `$hist_TargetPop` to `$TargetPop`

Mark:

SERV.TargetPop

Description:

Target Population Changed - Target Population value for Service Unit differs between historical and current data

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       SERV.TargetPop,
       hist_SERV.TargetPop as hist_TargetPop
from SERV
join hist.SERV as hist_SERV using(State, RegId, OrgId, ClusId, SUIId)
where SERV.TargetPop != hist_SERV.TargetPop
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)

- [State](#)
- [TargetPop](#)

4.1.88. ServTargetPopMissing

Class:

Missing

Priority:

High

Message:

Missing data - TargetPop `$TargetPop.q`

Mark:

SERV.TargetPop

Description:

Missing data - Target Population (TargetPop)

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       TargetPop
from SERV
where TargetPop is null
```

Data Elements:

- [ClusId](#)
- [OrgId](#)
- [RegId](#)
- [SUIId](#)
- [State](#)
- [TargetPop](#)

4.1.89. StConCount

SQL:

```
select sum(ServConCountTotal) as StConCountTotal
from ServConCount
```

4.1.90. StConDxPrincMissingHighProp

Class:

Historical

Priority:

High

Message:

DxPrinc Missing codes, as a proportional of all records, have increased by \$prop % from the previous year's submission. (\$h_prop % to \$n_prop %)

Mark:

HR.State

Description:

DxPrinc Missing codes have increased as a proportion of all records by 10% or more from the previous year's submission.

SQL:

```
with DxPrincMissingCounts as (
    select State,
           count(*) as n_con,
           sum(case when DxPrinc is null then 1 else 0 end) as n_missing
    from CON
    group by State
),
HistDxPrincMissingCounts as (
    select State,
           count(*) as h_con,
           sum(case when DxPrinc is null then 1 else 0 end) as h_missing
    from hist.CON
    group by State
) select State,
round(100.0 * h_missing / h_con, 1) as h_prop,
round(100.0 * n_missing / n_con, 1) as n_prop,
round((100.0 * n_missing / n_con) - (100.0 * h_missing / h_con), 1) as prop
from DxPrincMissingCounts
join HistDxPrincMissingCounts using (State)
where h_con > 0
and n_con > 0
and (100.0 * n_missing / n_con) - (100.0 * h_missing / h_con) >= 10.0
```

Data Elements:

- DxPrinc
- State

4.1.91. StConGrowthVaries**Class:**

Historical

Priority:

Medium

Message:

Growth variation over 20% (\$Growth.perc) in total contact count

Mark:

HR

Description:

The total number of contacts has changed by more than 20 per cent from the previous year.

SQL:

```
select State,
       round(Growth, 3) as Growth
from HrConCountGrowth
where abs(Growth) > 0.20
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrConCountGrowth](#)

4.1.92. StContDurGrowthVaries

Class:

Historical

Priority:

Medium

Message:

Growth variation over 15% ([\\$Growth.perc](#)) in total contact duration

Mark:

HR

Description:

The total contact hours has changed by more than 15 per cent from the previous year.

SQL:

```
select State,
       round(Growth, 3) as Growth
from HrContDurTotalGrowth
where abs(Growth) > 0.15
```

Data Elements:

- [State](#)

Virtual Elements:

- [HrContDurTotalGrowth](#)

4.1.93. StHighConProp

Class:

Exceptional

Priority:

High

Message:

Total proportion of F99 as principle dx is greater than 30% for the jurisdiction ([\\$Prop.perc](#))

Mark:

HR.State

Description:

Proportion of principle dx = F99 is more than 30% for the jurisdiction

SQL:

```
SELECT dxprinc_total.State,
       ConCount,
       ConDxPrincCount,
       sd_div_safe(ConDxPrincCount, ConCount, 3) as Prop
FROM (
    select count(*) as ConCount
    from CON
    join HR using (State)
) state_total,
(
    select State,
           count(*) as ConDxPrincCount,
           DxPrinc
    from CON
    join HR using (State)
    group by State,
             DxPrinc
) dxprinc_total
where DxPrinc = 'F99'
and sd_div_safe(ConDxPrincCount, ConCount, 3) > 0.3
```

Data Elements:

- DxPrinc
- State

4.1.94. SussHrDoBCount

Class:

Anomaly

Priority:

Low

Message:

More than double the average birthrate (\$DoBCount vs \$PerDoBCountAvg) on a suspicious date
(\$DoB.ddmmyyy)

Mark:

HR.State

Description:

There are greater than double the average number of births for a date which is likely to be a default or erroneous, selected from 9/9/9, 9/9/99, 1/1/1970, 1/1/11. Applied to “accurate” DoBFlag dates only.

SQL:

```
select State,
       DoB,
       count(*) AS DoBCount,
       round(PerDoBCountAvg,1) as PerDoBCountAvg
from PER,
     PerDoBCount
where DoB in ('1970-01-01', '1911-01-01', '2011-01-01', '1909-09-09', '1999-09-09')
      AND PersIdFlag = '1'
      AND DoBFlag = '1'
group by State,
       DoB,
       PerDoBCountAvg
having count(*) > PerDoBCountAvg * 2
```

Data Elements:

- [DoB](#)
- [DoBFlag](#)
- [PersIdFlag](#)
- [State](#)

4.2. Virtual Elements

4.2.1. ConAge

Base:

CON

Title:

Age at Contact

SQL:

```
select State,
       RegId,
       OrgId,
       ClusId,
       SUIId,
       PersIdFlag,
       PersId,
       RecordId,
       FLOOR((ContDt - DoB) / 365.25) as Age
from CON
join PER using (State, RegId, OrgId, ClusId, SUIId, PersIdFlag, PersId)
where DoB != '9999-09-09'
      and DoBFlag in ('1', '2')
```

Rules:

- [AdultAgeInYthOrCAUnitHigh](#)
- [AdultAgeInYthOrCAUnitLow](#)
- [BadDxPrincAd](#)
- [BadDxPrincLowAge](#)
- [BadDxPrincPpm](#)

- [HighAge](#)
- [LowAge](#)
- [LowAgeInOldUnitHigh](#)
- [LowAgeInOldUnitLow](#)
- [LowAgeMarriageHigh](#)
- [LowAgeMarriageLow](#)

4.2.2. HrCoBPropNotAus

Base:

HR

Title:

State Birth Country not Australia Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN CoB not in ('1100', '1101', '1102', '1199')
AND PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where CoB is not null
 group by State
```

Rules:

- [HighExtCoBProp](#)

4.2.3. HrCoBPropSuppRegistered

Base:

HR

Title:

State Birth Country is Supplementary

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN CoB ~ '^d{1,3} *' OR CoB LIKE '%00' AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where CoB is not null
 group by State
```

Rules:

- [HighSuppCoBProp](#)

4.2.4. HrConCount

Base:

HR

Title:

HR Count for Con

SQL:

```
select State,
       coalesce(Count, 0) as Count
from HR
left join (
  select State,
         count(*) as Count
  from CON
  group by State
) as foo using (State)
```

4.2.5. HrConCountChange

Base:

HR

Title:

HR Change in count for Con

SQL:

```
select State,
       (New.Count - Old.Count) as Change
from HrConCount as New
join hist.HrConCount as Old using (State)
```

4.2.6. HrConCountGrowth

Base:

HR

Title:

HR Growth in count for Con

SQL:

```
select State,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from HrConCount as New
join hist.HrConCount as Old using (State)
```

Rules:

- [StConGrowthVaries](#)

4.2.7. HrContDurChange

Base:

HR

Title:

HR Change in total contact duration

SQL:

```
select State,
       (New.Total - Old.Total) as Change
from HrContDurTotal as New
join hist.HrContDurTotal as Old using (State)
```

4.2.8. HrContDurTotal

Base:

HR

Title:

HR Total contact duration

SQL:

```
select State,
       coalesce(Total, 0) as Total
from HR
left join (
    select State,
           sum(ContDur) as Total
    from CON
    group by State
) as foo using (State)
```

4.2.9. HrContDurTotalGrowth

Base:

HR

Title:

HR Growth in total contact duration

SQL:

```
select State,
       sd_div_safe(New.Total - Old.Total, Old.Total, 3) as Growth
from HrContDurTotal as New
join hist.HrContDurTotal as Old using (State)
```

Rules:

- [StContDurGrowthVaries](#)

4.2.10. HrContParticProp

Base:

HR

Title:

State Participation Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN ContPartic != '1' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
  from CON
 where ContPartic is not null
 group by State
```

Rules:

- [ContParticChange](#)

4.2.11. HrDoBFlagPropRegistered

Base:

HR

Title:

State Estimated DoB Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN DoBFlag in ('2','8','9') AND PersIdFlag =
'1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where DoBFlag is not null
 group by State
```

Rules:

- [HighEstDoBFlagProp](#)

4.2.12. HrLegalStInvolCount

Base:

HR

Title:

HR Count for LegalSt Invol (1)

SQL:

```
select State,
       coalesce(Count, 0) as Count
  from HR
 left join (
    select State,
           count(*) as Count
      from CON
     where LegalSt = '1'
    group by State
  ) as foo using (State)
```

Rules:

- [BadHrLegalSt12Counts](#)

4.2.13. HrLegalStInvolRatio

Base:

HR

Title:

HR Ratio of LegalSt Invol (1)

SQL:

```
select State,
       sd_div_safe(HrLegalStInvolCount.Count, HrConCount.Count, 3) as Ratio
from HrLegalStInvolCount
join HrConCount using (State)
```

4.2.14. HrLegalStInvolRatioChange

Base:

HR

Title:

HR Change in ratio of LegalSt Invol

SQL:

```
select State,
       (New.Ratio - Old.Ratio) as Change
from HrLegalStInvolRatio as New
join hist.HrLegalStInvolRatio as Old using (State)
```

4.2.15. HrLegalStInvolRatioGrowth

Base:

HR

Title:

HR Growth in ratio of LegalSt Invol

SQL:

```
select State,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from HrLegalStInvolRatio as New
join hist.HrLegalStInvolRatio as Old using (State)
```

4.2.16. HrLegalStPropRegistered

Base:

HR

Title:

State Missing Legal Status Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN LegalSt = '9' AND PersIdFlag = '1' THEN 1
ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where LegalSt is not null
group by State
```

Rules:

- [HighMissingLegalStProp](#)

4.2.17. HrLegalStVolCount

Base:

HR

Title:

HR Count for LegalSt Vol (2)

SQL:

```
select State,
       coalesce(Count, 0) as Count
from HR
left join (
  select State,
         count(*) as Count
  from CON
  where LegalSt = '2'
  group by State
) as foo using (State)
```

Rules:

- [BadHrLegalSt12Counts](#)

4.2.18. HrLegalStVolRatio

Base:

HR

Title:

HR Ratio of LegalSt Vol (2)

SQL:

```
select State,
       sd_div_safe(HrLegalStVolCount.Count, HrConCount.Count, 3) as Ratio
from HrLegalStVolCount
join HrConCount using (State)
```

4.2.19. HrLegalStVolRatioChange

Base:

HR

Title:

HR Change in ratio of LegalSt Vol

SQL:

```
select State,
       (New.Ratio - Old.Ratio) as Change
from   HrLegalStVolRatio as New
join   hist.HrLegalStVolRatio as Old using (State)
```

4.2.20. HrLegalStVolRatioGrowth

Base:

HR

Title:

HR Growth in ratio of LegalSt Vol

SQL:

```
select State,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from   HrLegalStVolRatio as New
join   hist.HrLegalStVolRatio as Old using (State)
```

4.2.21. HrPerCount

Base:

HR

Title:

HR Count for Per

SQL:

```
select State,
       coalesce(Count, 0) as Count
from   HR
left join (
  select State,
         count(*) as Count
  from   PER
  group by State
) as foo using (State)
```

4.2.22. HrPerCountChange

Base:

HR

Title:

HR Change in count for Per

SQL:

```
select State,
       (New.Count - Old.Count) as Change
from   HrPerCount as New
join   hist.HrPerCount as Old using (State)
```

4.2.23. HrPerCountGrowth

Base:

HR

Title:

HR Growth in count for Per

SQL:

```
select State,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from HrPerCount as New
join hist.HrPerCount as Old using (State)
```

4.2.24. HrPersIdFlagDummyCount

Base:

HR

Title:

HR Count for PersIdFlag Dummy (2)

SQL:

```
select State,
       coalesce(Count, 0) as Count
from HR
left join (
    select State,
           count(*) as Count
    from PER
    where PersIdFlag = '2'
    group by State
) as foo using (State)
```

Rules:

- [BadHrPersIdFlag21Counts](#)

4.2.25. HrPersIdFlagDummyRatio

Base:

HR

Title:

HR Ratio of PersIdFlag Dummy (2)

SQL:

```
select State,
       sd_div_safe(HrPersIdFlagDummyCount.Count, HrPerCount.Count, 3) as Ratio
from HrPersIdFlagDummyCount
join HrPerCount using (State)
```

4.2.26. HrPersIdFlagDummyRatioChange

Base:

HR

Title:

HR Change in ratio of PersIdFlag Dummy

SQL:

```
select State,
       (New.Ratio - Old.Ratio) as Change
  from HrPersIdFlagDummyRatio as New
 join hist.HrPersIdFlagDummyRatio as Old using (State)
```

4.2.27. HrPersIdFlagDummyRatioGrowth

Base:

HR

Title:

HR Growth in ratio of PersIdFlag Dummy

SQL:

```
select State,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
  from HrPersIdFlagDummyRatio as New
 join hist.HrPersIdFlagDummyRatio as Old using (State)
```

4.2.28. HrPersIdFlagProp

Base:

HR

Title:

State Dummy PersId Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN PersIdFlag = '2' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
  from PER
 where PersIdFlag is not null
 group by State
```

Rules:

- [HighPersIdFlagProp](#)

4.2.29. HrPersIdFlagRealCount

Base:

HR

Title:

HR Count for PersIdFlag Real (1)

SQL:

```
select State,
       coalesce(Count, 0) as Count
from HR
left join (
    select State,
           count(*) as Count
    from PER
    where PersIdFlag = '1'
    group by State
) as foo using (State)
```

Rules:

- [BadHrPersIdFlag21Counts](#)

4.2.30. HrPersIdFlagRealRatio

Base:

HR

Title:

HR Ratio of PersIdFlag Real (1)

SQL:

```
select State,
       sd_div_safe(HrPersIdFlagRealCount.Count, HrPerCount.Count, 3) as Ratio
from HrPersIdFlagRealCount
join HrPerCount using (State)
```

4.2.31. HrPersIdFlagRealRatioChange

Base:

HR

Title:

HR Change in ratio of PersIdFlag Real

SQL:

```
select State,
       (New.Ratio - Old.Ratio) as Change
from HrPersIdFlagRealRatio as New
join hist.HrPersIdFlagRealRatio as Old using (State)
```

4.2.32. HrPersIdFlagRealRatioGrowth

Base:

HR

Title:

HR Growth in ratio of PersIdFlag Real

SQL:

```
select State,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from HrPersIdFlagRealRatio as New
join hist.HrPersIdFlagRealRatio as Old using (State)
```

4.2.33. HrResAreaProp

Base:

HR

Title:

State in-state SA2 Proportion

SQL:

```
select State,
       sd_div_safe(coalesce(sum(CASE WHEN State != substr(ResArea, 1, 1) AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where ResArea is not null
group by State
```

Rules:

- [HighExtSA2Prop](#)

4.2.34. OrgCoBPropNotAus

Base:

ORG

Title:

Organisation Birth Country not Australia Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN CoB not in ('1100', '1101', '1102', '1199')
AND PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from PER
where CoB is not null
group by State,
       RegId,
       OrgId
```

4.2.35. OrgCoBPropSuppRegistered

Base:

ORG

Title:

Organisation Birth Country is Supplementary

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN CoB ~ '^d{1,3} *' OR CoB LIKE '%00' AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from PER
where CoB is not null
group by State,
       RegId,
       OrgId
```

4.2.36. OrgConCount

Base:

ORG

Title:

ORG Count for Con

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
  select State,
         RegId,
         OrgId,
         count(*) as Count
  from CON
  group by State,
         RegId,
         OrgId
) as foo using (State, RegId, OrgId)
```

4.2.37. OrgConCountChange

Base:

ORG

Title:

ORG Change in count for Con

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Count - Old.Count) as Change
from OrgConCount as New
join hist.OrgConCount as Old using (State, RegId, OrgId)
```

4.2.38. OrgConCountGrowth

Base:

ORG

Title:

ORG Growth in count for Con

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from OrgConCount as New
join hist.OrgConCount as Old using (State, RegId, OrgId)
```

4.2.39. OrgContDurChange

Base:

ORG

Title:

ORG Change in total contact duration

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Total - Old.Total) as Change
from OrgContDurTotal as New
join hist.OrgContDurTotal as Old using (State, RegId, OrgId)
```

4.2.40. OrgContDurTotal

Base:

ORG

Title:

ORG Total contact duration

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Total, 0) as Total
from ORG
left join (
    select State,
           RegId,
           OrgId,
           sum(ContDur) as Total
    from CON
    group by State,
             RegId,
             OrgId
) as foo using (State, RegId, OrgId)
```


4.2.41. OrgContDurTotalGrowth

Base:

ORG

Title:

ORG Growth in total contact duration

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Total - Old.Total, Old.Total, 3) as Growth
from OrgContDurTotal as New
join hist.OrgContDurTotal as Old using (State, RegId, OrgId)
```

4.2.42. OrgContParticProp

Base:

ORG

Title:

Organisation Participartion Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN ContPartic != '1' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
from CON
where ContPartic is not null
group by State,
       RegId,
       OrgId
```

4.2.43. OrgDoBFlagPropRegistered

Base:

ORG

Title:

Organisation Estimated DoB Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN DoBFlag in ('2','8','9') AND PersIdFlag =
'1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from PER
where DoBFlag is not null
group by State,
       RegId,
       OrgId
```

4.2.44. OrgHasServCA

Base:

ORG

Title:

SERV Child and Adolescent below ORG

SQL:

```
select State,
        RegId,
        OrgId,
        Count
  from OrgServCACount
 where Count > 0
```

4.2.45. OrgHasServFor

Base:

ORG

Title:

SERV Forensic below ORG

SQL:

```
select State,
        RegId,
        OrgId,
        Count
  from OrgServForCount
 where Count > 0
```

4.2.46. OrgHasServGen

Base:

ORG

Title:

SERV General below ORG

SQL:

```
select State,
        RegId,
        OrgId,
        Count
  from OrgServGenCount
 where Count > 0
```

4.2.47. OrgHasServOld

Base:

ORG

Title:

SERV Older person below ORG

SQL:

```
select State,
       RegId,
       OrgId,
       Count
  from OrgServOldCount
 where Count > 0
```

4.2.48. OrgHasServYth

Base:

ORG

Title:

SERV Youth below ORG

SQL:

```
select State,
       RegId,
       OrgId,
       Count
  from OrgServYthCount
 where Count > 0
```

4.2.49. OrgIndigStPropNonAus

Base:

ORG

Title:

Organisation Indigenous born outside Australia Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sum(CoB not in ('1100','1101','1102','1199'))::INT) / count(*)::FLOAT as prop
  from PER
 where IndigSt in ('1','2','3')
       and CoB is not null
 group by State,
          RegId,
          OrgId
```

Rules:

- [HighIndigNonAustProp](#)

4.2.50. OrgLegalStInvolCount

Base:

ORG

Title:

ORG Count for LegalSt Invol (1)

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from CON
    where LegalSt = '1'
    group by State,
             RegId,
             OrgId
) as foo using (State, RegId, OrgId)
```

4.2.51. OrgLegalStInvolRatio

Base:

ORG

Title:

ORG Ratio of LegalSt Invol (1)

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(OrgLegalStInvolCount.Count, OrgConCount.Count, 3) as Ratio
from OrgLegalStInvolCount
join OrgConCount using (State, RegId, OrgId)
```

4.2.52. OrgLegalStInvolRatioChange

Base:

ORG

Title:

ORG Change in ratio of LegalSt Invol

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Ratio - Old.Ratio) as Change
from OrgLegalStInvolRatio as New
join hist.OrgLegalStInvolRatio as Old using (State, RegId, OrgId)
```

4.2.53. OrgLegalStInvolRatioGrowth

Base:

ORG

Title:

ORG Growth in ratio of LegalSt Invol

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from OrgLegalStInvolRatio as New
join hist.OrgLegalStInvolRatio as Old using (State, RegId, OrgId)
```

Rules:

- [OrgInvolGrowthVaries](#)

4.2.54. OrgLegalStPropRegistered

Base:

ORG

Title:

Organisation Missing Legal Status Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN LegalSt = '9' AND PersIdFlag = '1' THEN 1
ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where LegalSt is not null
group by State,
       RegId,
       OrgId
```

4.2.55. OrgLegalStVolCount

Base:

ORG

Title:

ORG Count for LegalSt Vol (2)

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from CON
    where LegalSt = '2'
    group by State,
           RegId,
           OrgId
) as foo using (State, RegId, OrgId)
```

4.2.56. OrgLegalStVolRatio

Base:

ORG

Title:

ORG Ratio of LegalSt Vol (2)

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(OrgLegalStVolCount.Count, OrgConCount.Count, 3) as Ratio
from OrgLegalStVolCount
join OrgConCount using (State, RegId, OrgId)
```

4.2.57. OrgLegalStVolRatioChange

Base:

ORG

Title:

ORG Change in ratio of LegalSt Vol

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Ratio - Old.Ratio) as Change
from OrgLegalStVolRatio as New
join hist.OrgLegalStVolRatio as Old using (State, RegId, OrgId)
```

4.2.58. OrgLegalStVolRatioGrowth

Base:

ORG

Title:

ORG Growth in ratio of LegalSt Vol

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from OrgLegalStVolRatio as New
join hist.OrgLegalStVolRatio as Old using (State, RegId, OrgId)
```

4.2.59. OrgPerCount

Base:

ORG

Title:

ORG Count for Per

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from PER
    group by State,
             RegId,
             OrgId
) as foo using (State, RegId, OrgId)
```

4.2.60. OrgPerCountChange

Base:

ORG

Title:

ORG Change in count for Per

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Count - Old.Count) as Change
from OrgPerCount as New
join hist.OrgPerCount as Old using (State, RegId, OrgId)
```

4.2.61. OrgPerCountGrowth

Base:

ORG

Title:

ORG Growth in count for Per

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from OrgPerCount as New
join hist.OrgPerCount as Old using (State, RegId, OrgId)
```

4.2.62. OrgPersIdFlagDummyCount

Base:

ORG

Title:

ORG Count for PersIdFlag Dummy (2)

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from PER
    where PersIdFlag = '2'
    group by State,
             RegId,
             OrgId
) as foo using (State, RegId, OrgId)
```

4.2.63. OrgPersIdFlagDummyRatio

Base:

ORG

Title:

ORG Ratio of PersIdFlag Dummy (2)

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(OrgPersIdFlagDummyCount.Count, OrgPerCount.Count, 3) as Ratio
from OrgPersIdFlagDummyCount
join OrgPerCount using (State, RegId, OrgId)
```


4.2.64. OrgPersIdFlagDummyRatioChange

Base:

ORG

Title:

ORG Change in ratio of PersIdFlag Dummy

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Ratio - Old.Ratio) as Change
from OrgPersIdFlagDummyRatio as New
join hist.OrgPersIdFlagDummyRatio as Old using (State, RegId, OrgId)
```

4.2.65. OrgPersIdFlagDummyRatioGrowth

Base:

ORG

Title:

ORG Growth in ratio of PersIdFlag Dummy

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from OrgPersIdFlagDummyRatio as New
join hist.OrgPersIdFlagDummyRatio as Old using (State, RegId, OrgId)
```

4.2.66. OrgPersIdFlagProp

Base:

ORG

Title:

Organisation Dummy PersId Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN PersIdFlag = '2' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
from PER
where PersIdFlag is not null
group by State,
       RegId,
       OrgId
```

4.2.67. OrgPersIdFlagRealCount

Base:

ORG

Title:

ORG Count for PersIdFlag Real (1)

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
  select State,
         RegId,
         OrgId,
         count(*) as Count
  from PER
  where PersIdFlag = '1'
  group by State,
         RegId,
         OrgId
) as foo using (State, RegId, OrgId)
```

4.2.68. OrgPersIdFlagRealRatio

Base:

ORG

Title:

ORG Ratio of PersIdFlag Real (1)

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(OrgPersIdFlagRealCount.Count, OrgPerCount.Count, 3) as Ratio
from OrgPersIdFlagRealCount
join OrgPerCount using (State, RegId, OrgId)
```

4.2.69. OrgPersIdFlagRealRatioChange

Base:

ORG

Title:

ORG Change in ratio of PersIdFlag Real

SQL:

```
select State,
       RegId,
       OrgId,
       (New.Ratio - Old.Ratio) as Change
from OrgPersIdFlagRealRatio as New
join hist.OrgPersIdFlagRealRatio as Old using (State, RegId, OrgId)
```

4.2.70. OrgPersIdFlagRealRatioGrowth

Base:

ORG

Title:

ORG Growth in ratio of PersIdFlag Real

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from OrgPersIdFlagRealRatio as New
join hist.OrgPersIdFlagRealRatio as Old using (State, RegId, OrgId)
```

4.2.71. OrgResAreaProp

Base:

ORG

Title:

Organisation in-state SA2 Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN State != substr(ResArea, 1, 1) AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where ResArea is not null
group by State,
       RegId,
       OrgId
```

4.2.72. OrgServCACount

Base:

ORG

Title:

SERV Child and Adolescent Count at ORG Level

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from SERV
    where TargetPop = '1'
    group by State,
             RegId,
             OrgId
) as tmpinner using (State, RegId, OrgId)
```

4.2.73. OrgServForCount

Base:

ORG

Title:

SERV Forensic Count at ORG Level

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from SERV
    where TargetPop = '3'
    group by State,
             RegId,
             OrgId
) as tmpinner using (State, RegId, OrgId)
```

4.2.74. OrgServGenCount

Base:

ORG

Title:

SERV General Count at ORG Level

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from SERV
    where TargetPop = '4'
    group by State,
           RegId,
           OrgId
) as tmpinner using (State, RegId, OrgId)
```

4.2.75. OrgServOldCount

Base:

ORG

Title:

SERV Older person Count at ORG Level

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from SERV
    where TargetPop = '2'
    group by State,
           RegId,
           OrgId
) as tmpinner using (State, RegId, OrgId)
```

4.2.76. OrgServYthCount

Base:

ORG

Title:

SERV Youth Count at ORG Level

SQL:

```
select State,
       RegId,
       OrgId,
       coalesce(Count, 0) as Count
from ORG
left join (
    select State,
           RegId,
           OrgId,
           count(*) as Count
    from SERV
    where TargetPop = '5'
    group by State,
           RegId,
           OrgId
) as tmpinner using (State, RegId, OrgId)
```

4.2.77. OrgUnknownContParticProp

Base:

ORG

Title:

Organisation Participartion Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN ContPartic = '8' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
from CON
where ContPartic is not null
group by State,
       RegId,
       OrgId
```

Rules:

- [HighUnknownContParticProp](#)

4.2.78. OrgUnknownContSessTypeProp

Base:

ORG

Title:

Organisation Session Type Proportion

SQL:

```
select State,
       RegId,
       OrgId,
       sd_div_safe(coalesce(sum(CASE WHEN ContSesType = '8' THEN 1 ELSE 0 END),
0.0), count(*), 3) as prop
  from CON
 where ContSesType is not null
 group by State,
         RegId,
         OrgId
```

Rules:

- [HighUnknownContSesTypeProp](#)

4.2.79. RegCoBPropNotAus

Base:

REG

Title:

Region Birth Country not Australia Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN CoB not in ('1100','1101','1102','1199')
AND PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where CoB is not null
 group by State,
         RegId
```

4.2.80. RegCoBPropSuppRegistered

Base:

REG

Title:

Region Birth Country is Supplementary

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN CoB ~ '^d{1,3} *' OR CoB LIKE '%00' AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where CoB is not null
 group by State,
         RegId
```

4.2.81. RegConCount

Base:

REG

Title:

REG Count for Con

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
    select State,
           RegId,
           count(*) as Count
    from CON
    group by State,
           RegId
) as foo using (State, RegId)
```

4.2.82. RegConCountChange

Base:

REG

Title:

REG Change in count for Con

SQL:

```
select State,
       RegId,
       (New.Count - Old.Count) as Change
from RegConCount as New
join hist.RegConCount as Old using (State, RegId)
```

4.2.83. RegConCountGrowth

Base:

REG

Title:

REG Growth in count for Con

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from RegConCount as New
join hist.RegConCount as Old using (State, RegId)
```


4.2.84. RegContDurChange

Base:

REG

Title:

REG Change in total contact duration

SQL:

```
select State,
       RegId,
       (New.Total - Old.Total) as Change
from RegContDurTotal as New
join hist.RegContDurTotal as Old using (State, RegId)
```

4.2.85. RegContDurTotal

Base:

REG

Title:

REG Total contact duration

SQL:

```
select State,
       RegId,
       coalesce(Total, 0) as Total
from REG
left join (
    select State,
           RegId,
           sum(ContDur) as Total
    from CON
    group by State,
             RegId
) as foo using (State, RegId)
```

4.2.86. RegContDurTotalGrowth

Base:

REG

Title:

REG Growth in total contact duration

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Total - Old.Total, Old.Total, 3) as Growth
from RegContDurTotal as New
join hist.RegContDurTotal as Old using (State, RegId)
```

4.2.87. RegContParticProp

Base:

REG

Title:

Region Participartion Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN ContPartic != '1' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
  from CON
 where ContPartic is not null
 group by State,
        RegId
```

4.2.88. RegDoBFlagPropRegistered

Base:

REG

Title:

Region Estimated DoB Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN DoBFlag in ('2','8','9') AND PersIdFlag =
'1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
  from PER
 where DoBFlag is not null
 group by State,
        RegId
```

4.2.89. RegIndigStPropNonAus

Base:

REG

Title:

Region Indigenous Born Outside Australia Proportion

SQL:

```
select State,
       RegId,
       sum(CoB not in ('1100','1101','1102','1199'))::INT / count(*)::FLOAT as prop
  from PER
 where IndigSt in ('1','2','3')
       and CoB is not null
 group by State,
        RegId
```

Rules:

- [RegIndigStGrowthVaries](#)

4.2.90. RegLegalStInvolCount

Base:

REG

Title:

REG Count for LegalSt Invol (1)

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
  select State,
         RegId,
         count(*) as Count
  from CON
  where LegalSt = '1'
  group by State,
         RegId
) as foo using (State, RegId)
```

4.2.91. RegLegalStInvolRatio

Base:

REG

Title:

REG Ratio of LegalSt Invol (1)

SQL:

```
select State,
       RegId,
       sd_div_safe(RegLegalStInvolCount.Count, RegConCount.Count, 3) as Ratio
from RegLegalStInvolCount
join RegConCount using (State, RegId)
```

4.2.92. RegLegalStInvolRatioChange

Base:

REG

Title:

REG Change in ratio of LegalSt Invol

SQL:

```
select State,
       RegId,
       (New.Ratio - Old.Ratio) as Change
from RegLegalStInvolRatio as New
join hist.RegLegalStInvolRatio as Old using (State, RegId)
```

4.2.93. RegLegalStInvolRatioGrowth

Base:

REG

Title:

REG Growth in ratio of LegalSt Invol

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from RegLegalStInvolRatio as New
join hist.RegLegalStInvolRatio as Old using (State, RegId)
```

4.2.94. RegLegalStPropRegistered

Base:

REG

Title:

Region Missing Legal Status Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN LegalSt = '9' AND PersIdFlag = '1' THEN 1
ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where LegalSt is not null
group by State,
       RegId
```

4.2.95. RegLegalStVolCount

Base:

REG

Title:

REG Count for LegalSt Vol (2)

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
  select State,
         RegId,
         count(*) as Count
  from CON
  where LegalSt = '2'
  group by State,
         RegId
) as foo using (State, RegId)
```

4.2.96. RegLegalStVolRatio

Base:

REG

Title:

REG Ratio of LegalSt Vol (2)

SQL:

```
select State,
       RegId,
       sd_div_safe(RegLegalStVolCount.Count, RegConCount.Count, 3) as Ratio
from RegLegalStVolCount
join RegConCount using (State, RegId)
```

4.2.97. RegLegalStVolRatioChange

Base:

REG

Title:

REG Change in ratio of LegalSt Vol

SQL:

```
select State,
       RegId,
       (New.Ratio - Old.Ratio) as Change
from RegLegalStVolRatio as New
join hist.RegLegalStVolRatio as Old using (State, RegId)
```

4.2.98. RegLegalStVolRatioGrowth

Base:

REG

Title:

REG Growth in ratio of LegalSt Vol

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from RegLegalStVolRatio as New
join hist.RegLegalStVolRatio as Old using (State, RegId)
```

4.2.99. RegPerCount

Base:

REG

Title:

REG Count for Per

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
  select State,
         RegId,
         count(*) as Count
  from PER
  group by State,
         RegId
) as foo using (State, RegId)
```

4.2.100. RegPerCountChange

Base:

REG

Title:

REG Change in count for Per

SQL:

```
select State,
       RegId,
       (New.Count - Old.Count) as Change
from RegPerCount as New
join hist.RegPerCount as Old using (State, RegId)
```

4.2.101. RegPerCountGrowth

Base:

REG

Title:

REG Growth in count for Per

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Count - Old.Count, Old.Count, 3) as Growth
from RegPerCount as New
join hist.RegPerCount as Old using (State, RegId)
```

4.2.102. RegPersIdFlagDummyCount

Base:

REG

Title:

REG Count for PersIdFlag Dummy (2)

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
  select State,
         RegId,
         count(*) as Count
  from PER
  where PersIdFlag = '2'
  group by State,
         RegId
) as foo using (State, RegId)
```

4.2.103. RegPersIdFlagDummyRatio

Base:

REG

Title:

REG Ratio of PersIdFlag Dummy (2)

SQL:

```
select State,
       RegId,
       sd_div_safe(RegPersIdFlagDummyCount.Count, RegPerCount.Count, 3) as Ratio
from RegPersIdFlagDummyCount
join RegPerCount using (State, RegId)
```

4.2.104. RegPersIdFlagDummyRatioChange

Base:

REG

Title:

REG Change in ratio of PersIdFlag Dummy

SQL:

```
select State,
       RegId,
       (New.Ratio - Old.Ratio) as Change
from RegPersIdFlagDummyRatio as New
join hist.RegPersIdFlagDummyRatio as Old using (State, RegId)
```

4.2.105. RegPersIdFlagDummyRatioGrowth

Base:

REG

Title:

REG Growth in ratio of PersIdFlag Dummy

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from RegPersIdFlagDummyRatio as New
join hist.RegPersIdFlagDummyRatio as Old using (State, RegId)
```

4.2.106. RegPersIdFlagProp

Base:

REG

Title:

Region Dummy PersId Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN PersIdFlag = '2' THEN 1 ELSE 0 END), 0.0),
count(*), 3) as prop
from PER
where PersIdFlag is not null
group by State,
       RegId
```

4.2.107. RegPersIdFlagRealCount

Base:

REG

Title:

REG Count for PersIdFlag Real (1)

SQL:

```
select State,
       RegId,
       coalesce(Count, 0) as Count
from REG
left join (
  select State,
         RegId,
         count(*) as Count
  from PER
  where PersIdFlag = '1'
  group by State,
         RegId
) as foo using (State, RegId)
```

4.2.108. RegPersIdFlagRealRatio

Base:

REG

Title:

REG Ratio of PersIdFlag Real (1)

SQL:

```
select State,
       RegId,
       sd_div_safe(RegPersIdFlagRealCount.Count, RegPerCount.Count, 3) as Ratio
from RegPersIdFlagRealCount
join RegPerCount using (State, RegId)
```

4.2.109. RegPersIdFlagRealRatioChange

Base:

REG

Title:

REG Change in ratio of PersIdFlag Real

SQL:

```
select State,
       RegId,
       (New.Ratio - Old.Ratio) as Change
from RegPersIdFlagRealRatio as New
join hist.RegPersIdFlagRealRatio as Old using (State, RegId)
```

4.2.110. RegPersIdFlagRealRatioGrowth

Base:

REG

Title:

REG Growth in ratio of PersIdFlag Real

SQL:

```
select State,
       RegId,
       sd_div_safe(New.Ratio - Old.Ratio, Old.Ratio, 3) as Growth
from RegPersIdFlagRealRatio as New
join hist.RegPersIdFlagRealRatio as Old using (State, RegId)
```

4.2.111. RegResAreaProp

Base:

REG

Title:

Region in-state SA2 Proportion

SQL:

```
select State,
       RegId,
       sd_div_safe(coalesce(sum(CASE WHEN State != substr(ResArea, 1, 1) AND
PersIdFlag = '1' THEN 1 ELSE 0 END), 0.0), count(*), 3) as prop
from CON
where ResArea is not null
group by State,
       RegId
```