

SMDB Developer

API updates fall 2025 developer guide

Danmarks Statistik

October 10, 2025

Document Version: 1.0

Contents

Contents	2
1 Introduction	3
1.1 Purpose of this guide	3
1.2 Reason for the changes	3
2 Relevant info about Stofmisbrugsdatabasen	4
2.1 Endpoints	4
2.2 Schema Ids	4
2.3 Schema hierarchy	5
3 Implementation details	7
3.1 Wrapper	7
3.2 Retrieving all active cases	8
3.3 Deletion	9
3.4 ID based retrieval	10

Chapter 1

Introduction

We recommend reading this document in its entirety, since we aim to include a lot of relevant information that otherwise needs to be gathered from multiple sources.

1.1 Purpose of this guide

In the fall of 2025 the Stofmisbrugsdatasen (SMDB) APIs will be expanded with new operations. This document intends to introduce developers responsible for integrating with SMDB to the new API functionality and its intended use.

It is assumed that the reader is familiar with integrating with SMDB in general, using one of the 2 equivalent SOAP based APIs. This document does not contain all information needed to implement a new integration from scratch. It is a companion document to the code lists found on the [main SMDB information page](#) and the specifications found under SMDB on the [guide page for implementation](#).

1.2 Reason for the changes

We hope to solve at least two major concerns. First, it will no longer be possible to change the start date of an already registered Anmodning-schema, in a way that will change the format expected for the schema, it may become necessary for users to start over completely.

The second concern is the cumbersome task when a municipality or treatment facility changes between systems which integrate with SMDB. Currently this is handled by deleting all schemas for all active cases, and then recreating them in the new system, which is both a potentially large manual task and error-prone. This is the recommended method by SMDB, since there has not been a way to retrieve the cases/schemas besides looking up the CPR number, which the system would normally not know in advance.

We hope to solve these issues, in addition to some quality of life improvements for both developers and users.

The changes are non-destructive and does not change any existing functionality, besides internal corrections which may result in things like some edge cases of data which was previously accepted being denied due to improvements in validation rules.

Chapter 2

Relevant info about Stofmisbrugsdatabasen

2.1 Endpoints

New functionality is added, but all existing SOAP operations in our API will continue to work as they did previously. There are new operations in both the endpoint for retrieving data and sending data.

As always, our preproduction SOAP-based definitions can be seen in the definition files linked below once deployed.

V0:

SecureHentData Retrieve data [smdbwebservice/api/external/v0/securehentdata.svc](#)

SecureIndsaetData Send data [smdbwebservice/api/external/v0/secureindsaetdata.svc](#)

V1:

SecureHentData Retrieve data [smdbwebservice/api/external/v1/securehentdata.svc](#)

SecureIndsaetData Send data [smdbwebservice/api/external/v1/secureindsaetdata.svc](#)

We recommend everybody to use V1 of the API unless you have specific historical reasons not to. V0 uses a less exact version of the DTOs with key/value pairs for many fields, which tools are unable to autogenerate from the definition files.

2.2 Schema Ids

All schemas (represented by the DTOs in the API) in SMDB have an Id, unique for that schema type. They are found on every DTO retrieved from the *securehentdata* endpoint, and when successfully registering a new schema through our API, the returned *ServiceResponseDTO* will contain a *PrimaryKey* which is the Id for the created schema.

2.3 Schema hierarchy

To help with implementing functionality for effectively retrieving data from SMDb using the new operations, the internal relationship between the various schemas which make up a case should be made clear. Figure [Figure 2.1](#) shows a diagram of the currently active schemas, besides the citizen which is an implied entity, only defined by the CPR number in the AnmodningDTO and the AsiDTO.

A drug treatment case (also referred to as KontaktForløb in Danish) will have these schemas filled out, describing each major step in the treatment:

Anmodning	"Request". The initial request for treatment, which also contains the fields for how the treatment eventually ended. This is effectively the object for a case, with some very basic information on the person requesting treatment. To make this clear, a few cases often will only have this object/schema marking it's existence, if the person never shows up for the next steps.
Iværksættelse	"Initiation". The schema filled out when starting the treatment. This contains the bulk of the details for the case, such as the severity of the problem. The examination where this schema is filled out, must happen before actual treatments are started, since it will usually determine the treatments needed and offered. It is related to the Anmodning, and one of these are filled out for each case.
Tilbudstilknytning	"Treatment attachment". Each different treatment type and facility during the case handling will be registered as a separate Tilbudstilknytning. These are smaller schemas describing the type of treatment being offered and where. Before a case is closed, they can go through multiple treatments, and there will often be several of these registered for each case. They are related to the Iværksættelse above.
ASI	"Addiction Severity Index". A large schema with further details about the issues being treated. This schema is no longer needed, but can still be created for cases which started before 1. July 2019. It is not related directly to the main cases. The schema is rarely handled by external systems.
KvalHep	"Hepatitis C and treatment quality". An additional schema specifically for checking treatment quality and disease. The schema was deprecated for cases starting after 1. July 2025. Adding and retrieving this schema demands the NemLog-in/MitId role <i>Sundhedsfaglig</i> , indicating that the schema is usually filled in by different people (similar to a nurse) than the others parts of a case. It can only be added if an Iværksættelse exists, since it is effectively an expansion to it.

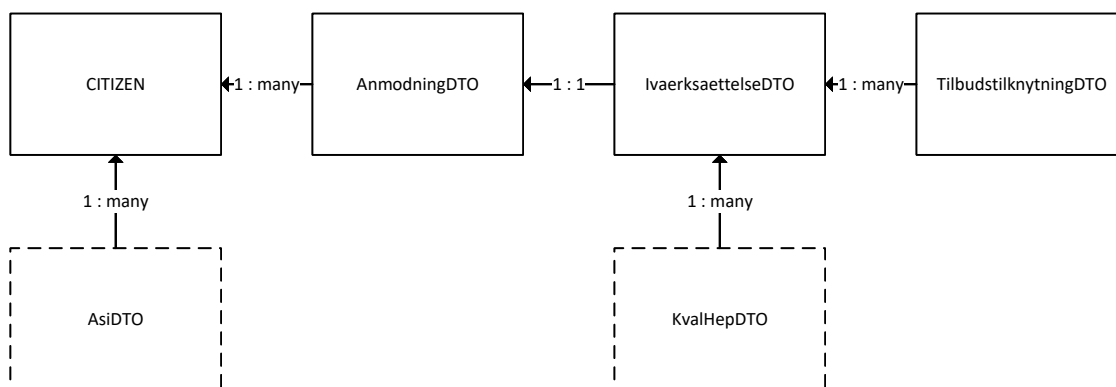


Figure 2.1: The relation between the various schemas. Dotted lines indicate schemas which are deprecated for all new cases. The names of the schemas are the ones used in the API.

Chapter 3

Implementation details

3.1 Wrapper

Retrieving data has been a bit perilous, since it returns a raw object, which means that there is no abstraction, and no distinction, between errors that are of a technical nature, and errors that can be shown to the user. The API for adding data has the `ServiceResponseDTO` object, which returns a status for the action. We are now introducing similar wrappers for a range of new SOAP operations, to make the distinction of error types easier, and to allow displaying the errors to the users. Due to technical limitations in SOAP, these wrappers are technically unique for each operation and have distinct names. They do follow a naming convention, `HentDataResultForSchemaDTO` with "Schema" being different for each. They also share all internal fields:

```
1 <name="SupportId" type="ser:guid"/>
2 <minOccurs="0" name="errorMessage" nillable="true" type="xs:string"/>
3 <name="hentDataStatus" type="HentDataStatus"/>
4 <minOccurs="0" name="resultDTO" nillable="true" type="
  HentDataResultForSchemaDTO"/>
```

The above XML is a slightly cleaned up version of what is found in the definition files linked to in section 2.1. `HentDataResultForSchemaDTO` shows the general naming scheme, "Schema" being different for each DTO.

All new SOAP operations for retrieving data will return an object like the above. Note that some of the elements are nullable, while others will always be present.

SupportId	A common Id which the SMDB team can use to check our internal log for exact error reasons or whatever else may be needed.
errorMessage	Always safe to display to users. Only exists in the case of errors.
resultDTO	The DTO being retrieved, if the <code>hentDataStatus</code> indicates a success.
hentDataStatus	An enum which is defined in the wsdl file. This will usually be the first thing you should look at, to determine which steps to take next.

The appropriate handling of the returned `HentDataResultForSchemaDTO` is determined by the `hentDataStatus` enum. The possible values are in Danish, so here follows a translation some comments on how to handle each value.

Ukendt	unknown. Unlikely to happen on retrieval, and generally indicates an unexpected internal error in the SMDB backend. This will not have an error message that the end user can react to.
Fejlet	Failed. An error not of the more normal types below. There will be an error message.
Vellykket	Success. The resultDTO will have the requested object. There will be no error message.
AdgangNægtet	Not allowed. The NemLog-in SAML assertions or the delegation lookup do not contain any needed permissions for data being requested.
DataErSlettet	Data is deleted. The data being requested has been marked as deleted. Recreation may be possible by contacting SMDB, but generally a new case must be created. Users are also able to mark data as deleted from the SMDB frontend directly.
DataIkkeTilgængelig	Data does not exist. This happens when trying to retrieve data using an ID which does not exist in SMDB.

A non-SOAP answer in pure html can now be expected to be variants of either connection or NemLog-in related errors.

3.2 Retrieving all active cases

To ease transition between systems we now allow retrieving a complete list of all active cases for a municipality. This will make it both easier and safer for a municipality to migrate to a new system, potentially saving them weeks of error prone data entry. From our perspective we hope the data will get fewer errors introduced during migrations.

The new operation is called **HentAktiveKontaktForloebIMyndighed** expects an integer called **kommuneKode**, which is a general Danish unique Id every municipality has. The operation will return a wrapper (see 3.1) object, containing a new DTO: **KontaktForloebArrayDTO**. As the name implies, this contains an array of `KontaktForloebDTO`, which each represents a basic case, and includes an `AnmodningDTO`, possibly an `IvaerksaettelseDTO`, and possibly an array of `TilbudstilknytningDTOs`.

Data access

As with all other SOAP operations, the least restricted NemLog-in/MitId role or delegation¹. If the external system (the readers system) uses a NemLog-in system user with delegations from multiple organizations/municipalities, the responsibility to only show the correct cases to each municipality/organization **must** be handled by the external

¹A delegation allows an organization to act on another organizations behalf, using specific permissions (such as `Indberetter` or `Sundhedsfaglig`).

system. Workers in a municipality is allowed to access all cases returned by this operation. If the system is used by a treatment facility, only cases created by users in that facility is allowed to be accessed by them. Since the cases do not currently carry information about this, **we recommend not using this functionality yet for treatment facilities.**

The reason only active cases can be retrieved using this method, is that only data relevant for reporting statistics back is legally allowed to be pulled from systems within Danmarks Statistik (DST).

3.3 Deletion

New operations in the *secureindsaetdata* endpoints will allow external systems to delete the schemas they are otherwise allowed to update. Since it will no longer be possible to change the start date of an already registered schema, in a way that will change the format expected for the schema, it may become necessary for users to start over completely.

All the operations take the Id (see 2.2) for the schema being deleted, except *SletHeltForloeb* as explained below.

SletAnmodning	Deletes the Anmodning. Only allowed if no Ivaerksaettelse is linked to it.
SletIvaerksaettelse	Deletes the Ivaerksaettelse. Only allowed if no Tilbudstilknytning or KvalHep is linked to it.
SletTilbudsTilknytning	Deletes the Tilbudstilknytning. Note that for a closed Anmodning, if any Tilbudstilknytning exists on the case, there are validation rules regarding how close the end dates are for the latest Tilbudstilknytning and the end date on the Anmodning. This means that users will usually have to reopen a case, in order to delete the latest Tilbudstilknytning.
SletKvalHep	Deletes a KvalHepDTO.
SletHeltForloeb	Deletes all schemas above for a case. It automatically deletes the schemas in the correct order. If an errors happens all changes are rolled back. The Id expected is for the Anmodning, which is the entry point schema for each case (see 2.3).

The deletion is not physical at first, as SMDB will retain of copy of the deleted data for a period. It will be ignored in all operations. When looking in the possible data retrieval wrapper (see 3.1) status values, deleted and missing data is handled differently. This means that attempting to retrieve a deleted schema, will change the status from deleted to missing after a period. The exact period is not yet determined.

3.4 ID based retrieval

The current way to retrieve data from SDMB is to ask for all schemas of a specific type, for a specific citizen in a specific municipality. The schemas do have a hierarchy, where some cannot exist without the previous schemas being registered and valid (see 2.3). This structure has until now only been reflected through the naming of ID fields between the related DTOs, and buried in the documentation, which has made the task of integrating properly with SMDDB a potentially daunting task. This is historically a consequence of the schemas not being as directly related in the data structure, and mainly based on relationships between dates on different schema types. We are aware that very few external systems currently use the functionality for retrieving the schemas owned by their users, and we expect this is mainly due to this unnecessary complexity.

Since the schemas can be created and modified other ways than just through the API, it does introduce some potential issues. The most obvious is unwittingly overwriting changes made through other channels, but also the difficulty in getting new schemas created outside the external system.

All the following new operations in the endpoints **securehentdata** take an Id and return a wrapper with the resultDTO inside (see 3.1):

HentAnmodning	Gets a specific AnmodningDTO from its Id.
HentIvaerksaettelse	Gets a specific IvaerksaettelseDTO from its Id.
HentIvaerksaettelseForAnmodning	Gets an IvaerksaettelseDTO from the Id of the related Anmodning.
HentTilbudstilknytning	Gets a specific TilbudstilknytningDTO from its Id.
HentTilbudstilknytningForIvaerksaettelse	Gets all TilbudstilknytningDTO from the Id of the related Ivaerksaettelse.
HentKvalHepMedId	Gets a specific KvalHepDTO from its Id. Notice that there is no current way to get this DTO from its related Ivaerksaettelse. It may be implemented in the future, but since the schema is being deprecated (see 2.3) further development is not currently planned.