Project No. 964220

Intelligent digital tools for screening of brain connectivity and dementia risk estimation in people affected by mild cognitive impairment

D2.2

Implementation of the Central Data Repository

WP 2 – Data Management and Features Extraction

Authors	DNV, Lurtis, UCM, IRCCS, UCSC, AALTO, HUS, Brainsymph
Lead participant	DNV
Delivery date	25th February 2022
Dissemination level	Public
Type	Report

Version 1

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 964220

Revision History

Author(s)	Description	Date
Abdillah Suyuthi, Abhilash Anand, Harry Hallock, Serena Marshall (DNV)	Draft deliverable (v1)	09/02/2022
Federico Ramírez Toraño (UCM)	Revision 1st draft	10/02/2022
Francesca Miraglia (IRCCS), Camillo Marra (UCSC)	Revision 1st draft	13/02/2022
Susanne Merz (AALTO)	Revision 1st draft	15/02/2022
Christoffer Hatlestad-Hall (Brainsymph)	Revision 1st draft	15/02/2022
Hanna Renvall (HUS)	Revision 1st draft	15/02/2022
Abdillah Suyuthi, Abhilash Anand, Harry Hallock (DNV)	Draft deliverable (v2)	16/02/2022
Victor Ayllón (Lurtis)	Revision 2nd draft	17/02/2022
Christoffer Hatlestad-Hall (Brainsymph)	Revision 2nd draft	18/02/2022
Camillo Marra (UCSC)	Revision 2nd draft	19/02/2022
Ricardo Bruña (UCM)	Revision 2nd draft	19/02/2022
Hanna Renvall (HUS)	Revision 2nd draft	20/02/2022
Abdillah Suyuthi, Abhilash Anand, Harry Hallock (DNV)	Draft deliverable (v3)	22/02/2022
Victor Ayllón (Lurtis)	Revision 3rd draft	22/02/2022
Abdillah Suyuthi, Abhilash Anand, Harry Hallock, Serena Marshall (DNV)	Final Version	23/02/2022
Andreia Cruz (accelCH)	Final Revision	23/02/2022
Lina Plataniti (OUS)	Final Revision	24/02/2022
Harry Hallock (DNV)	Final Version	25/02/2022

Abbreviations

CDR	Central Data Repository
CRC	Cyclic Redundancy Check
CRUD	Create, Read, Update, Delete (Operations)
DL	Data Lake
DW	Data Warehouse
EC	European Commission
ERD	Entity Relationship Diagram
EU	European Union
H2020	Horizon 2020
MCI	Mild Cognitive Impairment
MDR	Medical Device Regulation
ML/DL	Machine Learning/Deep Learning
PMS	Platform Management System
SQL	Structured Query Language
SRS	Software Requirements and Specifications
tbd	To Be Defined
TSD	Tjenester / Services for Sensitive Data
WP	Work Package

Partner Short Names

OUS	Oslo University Hospital
AALTO	Aalto University
accelCH	accelopment Schweiz AG
Brainsymph	BrainSymph AS
DNV	Det Norske Veritas
IRCCS	Scientific Institute for Research, Hospitalization and Healthcare, San Raffaele Roma
Lurtis	Lurtis Rules S.L
UCM	Complutense University of Madrid
UCSC	Università Cattolica del Sacro Cuore
HUS	Helsinki University Hospital

Table of Contents

REVISION HISTORY 2
ABBREVIATIONS 3
PARTNER SHORT NAMES 3
EXECUTIVE SUMMARY 6
Organisation of D2.2 6
1 INTRODUCTION 7
1.1 Background and motivation 7
1.2 Objectives 7
1.3 Scope 7
1.4 Interdependencies with other AI-Mind deliverables 8
2 CENTRAL DATA REPOSITORY OVERVIEW 9
2.1 Components, Functions and Data Managed 9
3 DATA LAKE DESIGN 10
3.1 Data Lake File System 10
3.2 Data Lake Database Design 10
3.2.1 Entity Relationship Diagram 11
3.2.2 Tables and Data Fields Specification 11
4 DATA WAREHOUSE DESIGN 12
4.1 Data Warehouse File System 13
4.2 Data Warehouse Database Design: Prospective 13
4.2.1 Entity Relationship Diagram 14
4.2.2 Tables and Data Fields Specification – Prospective Data 16
4.3 Data Warehouse Database Design: Retrospective - OUS 26
4.3.1 Entity Relationship Diagram 26
4.3.2 Tables and Data Fields Specification – Retrospective Data: OUS 26
4.4 Data Warehouse Database Design: Retrospective – UCM 26
4.4.1 Entity Relationship Diagram 27
4.4.2 Tables and Data Fields Specification – Retrospective Data: UCM 27
4.5 Data Warehouse Database Design: Retrospective - IRCCS/UCSC 27
4.5.1 Entity Relationship Diagram 28
4.5.2 Tables and Data Fields Specification – Retrospective Data: IRCCS/UCSC 28
5 IMPLEMENTATION 30
6 CONCLUSION 30
7 APPENDIX 31
7.1 Entity Relationship Diagram for Data Warehouse Database Model: Prospective 31

7.2 SQL Codes 32
7.2.1 SQL Code to Establish Tables related to Data Lake Database 32
7.2.2 SQL code to Establish Tables related to Prospective Data in Data Warehouse 32
7.2.3 SQL Code to Establish Tables related to Retrospective Data : OUS..... 37
7.2.4 SQL Code to Establish Tables related to Retrospective Data : UCM 38
7.2.5 SQL Code to Establish Tables related to Retrospective Data : IRCCS/UCSC 39

Executive Summary

AI-Mind's goal is to enable researchers and clinicians to perform an early assessment of the risk of developing dementia in people with mild cognitive impairment (MCI) using AI technology. As a software solution, it is fed with a diverse set of retrospective and prospective data. The data is ingested, transformed, and ultimately stored in a central data repository (CDR) responsible for ensuring persistence, consistency, concurrency, integrity, and security across all AI-Mind applications and processes.

This deliverable describes the implementation of AI-Mind's CDR, hosted on the TSD platform (Tjenester / Services for sensitive data). The report outlines the CDR's key functions and main components, mainly the data lake (DL) and data warehouse (DW), as well as the database designs and folder structures. More specifically, it details DL design, including file system and database, and also the DW design for raw data, including file system and database. It will serve as the basis for DW design for the remaining data types (i.e., standardised, curated etc), and the implementation of the CDR in the TSD Platform.

D2.2 continues the work done in D4.1 and will evolve throughout the project as more information about the requirements of the CDR are gathered.

Organisation of D2.2

Figure 1 presents the overview on how this report is organised.

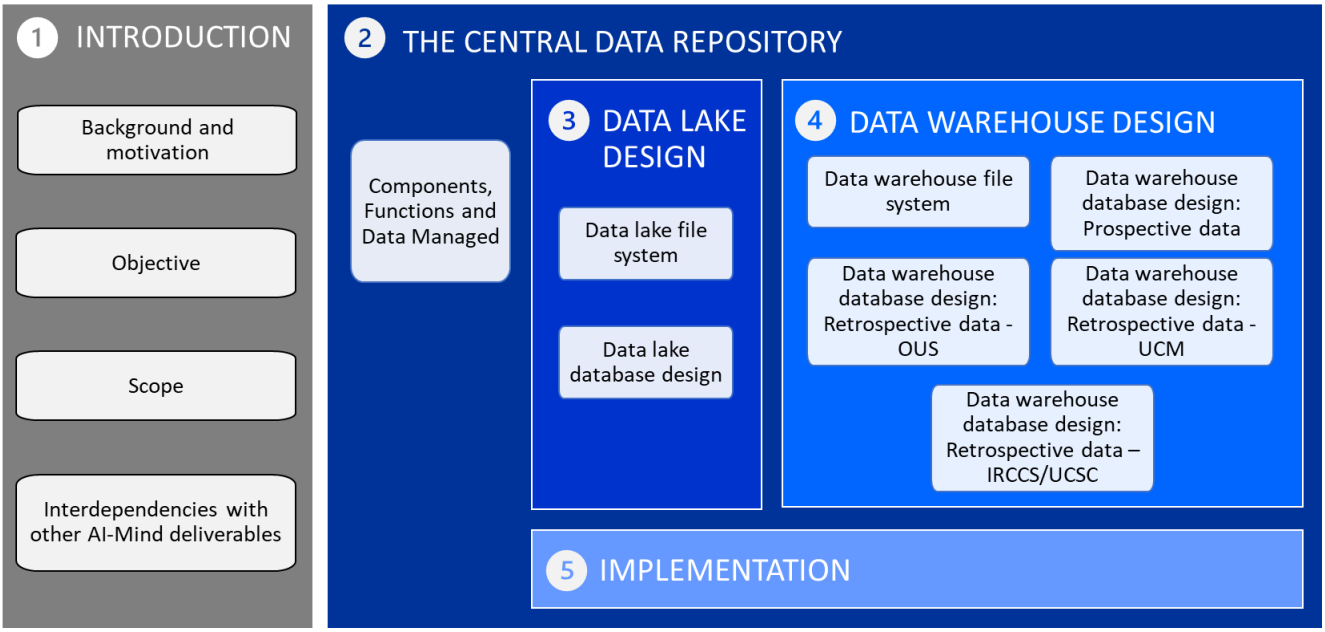

Figure 1. Organisation of deliverable 2.2

1 Introduction

1.1 Background and motivation

AI-Mind aims to develop tools that will support early diagnosis of mild cognitive impairment (MCI) and predict the risk of developing dementia. This pivots on the collection, storage, management, and processing of relevant high-quality data. The CDR plays a central role in the storage and management of all data in AI-Mind. AI-Mind's dependency on data means that the roles and functions of the CDR must be well-designed and robust.

1.2 Objectives

The objectives of this deliverable are to:

- Summarise the main components of the CDR
- Detail the DL design, including file system and database
- Detail the DW design for raw data, including file system and database
- Outline an implementation plan.

1.3 Scope

This report will focus on the lineage of the raw prospective and retrospective data within these environments, although acknowledgement is made to other components of the CDR, i.e., the DW file systems and databases for other types of data, as well as the code repository and the platform management system repository. **The scope of this report is indicated by the red outlined boxes in Figure 2.**
The other components are not discussed in this report, as enough detail is not known about them at the present time. Protocols for standardised and curated data are being developed currently in D2.3 - *Standardisation protocols for data collection and pre-processing*. **Components not covered will be discussed in updated versions of this report.**

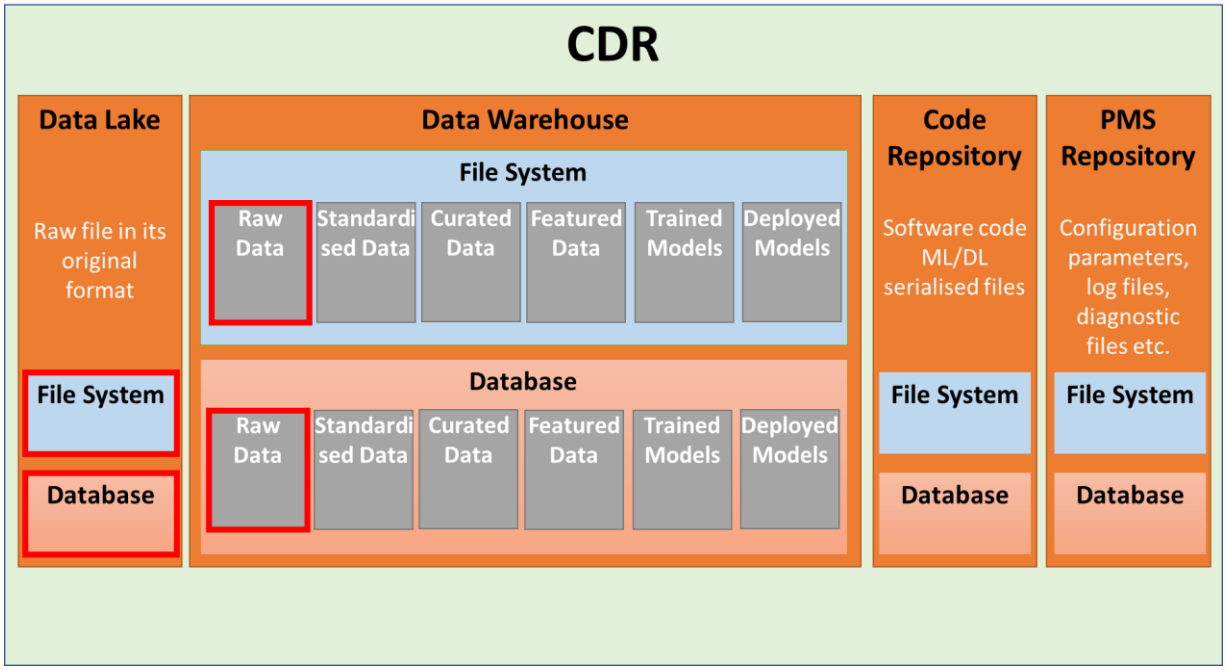

Figure 2. Components of the CDR based on requirements listed in D4.1. The outlined red boxes indicate what will be discussed in detail in this report. PMS = Platform Management System.

Due to DNV's designation as a Notified Body under the EU medical device regulation (MDR), and thus DNV's need to maintain impartiality, DNV will only be involved in the design and implementation of the following CDR components:

- DL File System
- DL Database
- DW File System – raw data
- DW Database – raw data
- DW File System – standardised data
- DW Database – standardised data

Additionally, DNV will develop a library for quality checking which will be implemented by certain partners at selected points of the data pipeline. The result of the quality checking will be saved in the PMS repository, where the platform administrator will monitor the process results in the different stages of the data pipeline.

1.4 Interdependencies with other AI-Mind deliverables

The implementation of the CDR within AI-Mind requires knowledge and expertise from other deliverables within the project. Thus, this deliverable will either provide input or benefit from the output of other deliverables.

The following have provided input for D2.2:

- D8.3 Project handbook (delivered M3)

- D2.1 Standardisation of available and prospective data collection (delivered M8)
- D1.3 Guidelines for data management and sharing (delivered M9)
- D5.1 Clinical study dossier (Clinical study protocol, Informed Consent Form) (delivered M9)
- D4.1 Software requirements & specifications (SRS) (delivered M10)
- D1.4 AI-Mind data governance and data management framework (delivered M12)

D2.2 will provide the data repository for many of the deliverables that come later in AI-Mind, allowing the development of the AI-Mind tools, including:

- D2.3 Standardisation protocols for data collection and pre-processing (to be delivered M16)
- D2.4 Standardisation of data integration and data repository management (to be delivered M37)
- D2.5 Updated standardisation protocols for data collection and pre-processing (to be delivered M38)
- D5.2 Midway and complete prospective dataset for the AI-Mind Connector and AI-Mind Predictor (to be delivered M24)
- D5.7 Complete Prospective data set for the AI-Mind Connector and AI-Mind Predictor (to be delivered M60)

2 Central Data Repository Overview

The CDR is the main working storage system of the project, responsible for hosting various types of data formats. As outlined in D4.1 Section 4.5, the CDR will ensure long-term persistence, consistency, and integrity of the data and code used in the project, including the PMS logs and configuration files.

2.1 Components, Functions and Data Managed

The CDR (see Figure 2) consists of:

1. Data lake (DL) - Provides storage for the raw data sets (in their original format) uploaded by each partner.
2. Data warehouse (DW) – the main storage and work area for the data pipeline (outlined in D4.1). Provides storage for the raw, standardised, and featured data sets, as well as the trained and deployed models.
3. Code Repository - Provides storage for software code repositories and ML/DL model's serialised files.
4. PMS Repository - Provides storage for all the data necessary to manage the internal AI-Mind platform, i.e., platform administration files (e.g., configuration parameters, log files, diagnostic files etc.)

3 Data Lake Design

The DL is a repository of data stored in its original format. Once the file is fetched from the file staging area, it will be moved to the DL file system and its metadata will be recorded into the DL database.

3.1 Data Lake File System

The DL file system is organised by data source, data category, and date (i.e., year and month), see Figure 3. A Python code will be written to set up the DL file system. Once deployed in TSD, it needs to run only once to prepare all necessary directories. The code may be designed to cover several years beyond the end of the AI-Mind project (expected February 2026). Alternatively, dynamic code may be written, which will automatically create more directories, according to whenever the file is moved.

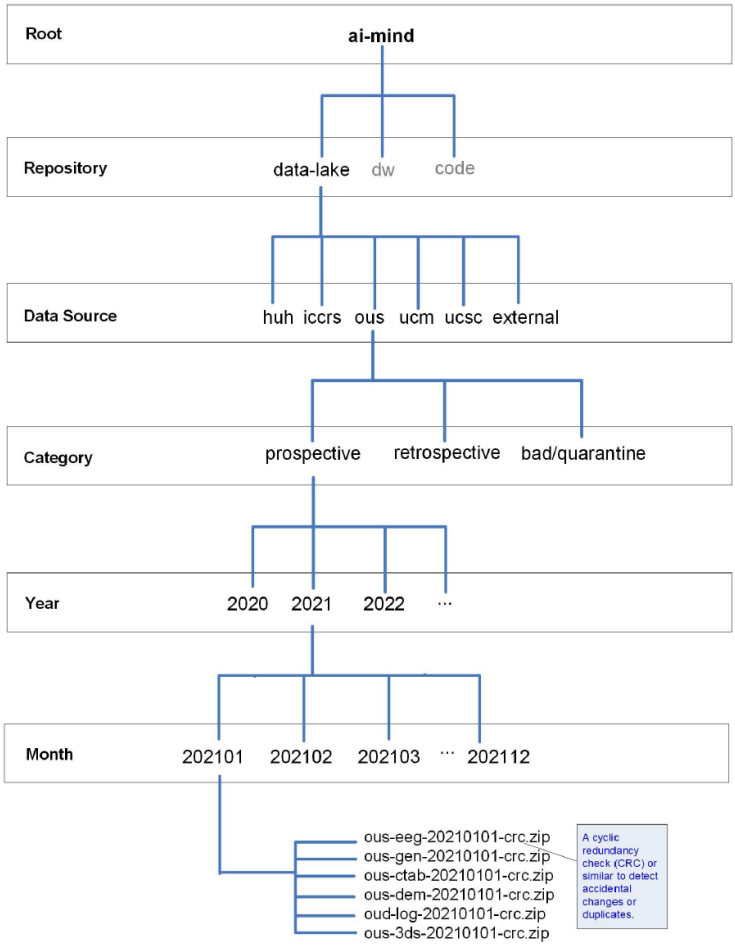

Figure 3 The Data Lake File System (Figure 12 in D4.1).

3.2 Data Lake Database Design

The main objective of the DL database is to store the catalogue and metadata of raw data ingested and stored in the DL file system. The entity relationship diagram (ERD), and corresponding tables and specifications of data fields, are described in the following sub-sections 3.2.1 and 3.2.2. The SQL code to establish these tables is presented in the Appendix (Section 7.2.1).

3.2.1 Entity Relationship Diagram

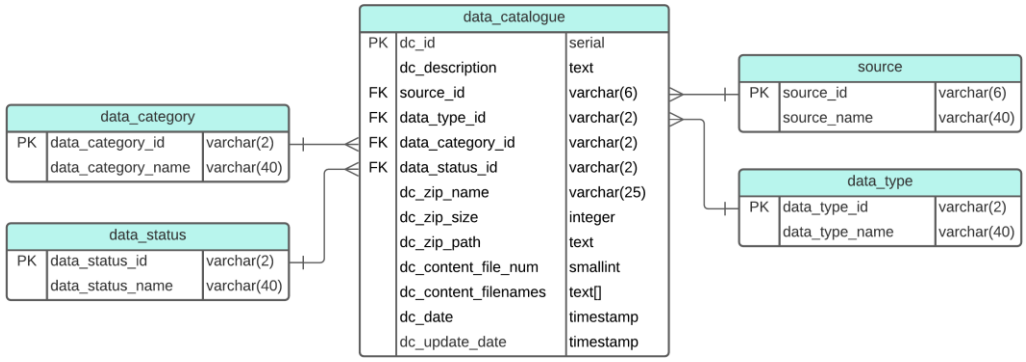

Figure 4 ERD for Data Lake metadata.

3.2.2 Tables and Data Fields Specification

Table 1 data_catalogue specification

Field Name	Description	Data Type	Unit	Note
dc_id	Data catalogue ID. Primary key.	Auto increment integer: serial		PK
dc_date	Date & time of file storage	Date and time: timestamp		
dc_description	Description of the file	String: text		
source_id	Source of the data file	String: varchar(6)		FK, link to table source
data_type_id	Type of data file	String: varchar(2)		FK, link to table data_type
data_category_id	Category of data file	String: varchar(2)		FK, link to table data_category
data_status_id	Status of data file	String: varchar(2)		FK, link to table data_status
dc_zip_name	Filename of the zip file	String: varchar(25)		
dc_zip_size	Size of the zip file	Integer	KB	
dc_zip_path	Path name where the zip file has been stored	String: text		
dc_content_ file_num	Number of files in the zip file	Integer: smallint		
dc_content_filenames	Name of files in the zip file	Array of string: text[]		
dc_update_date	Updated time and date of data file storage	Date and time: timestamp		

Table 2 source specification

Field Name	Description	Data Type	Unit	Note
source_id	Source ID. Primary key.	String: varchar(6)		PK
source_name	Source of the data file i.e., from OUS, UCM, IRCCS, HUS, UCSC, EXT	String: varchar(40)		

Table 3 data_type specification

Field Name	Description	Data Type	Unit	Note
data_type_id	Data type ID. Primary key.	String: varchar(2)		PK
data_type_name	Type of data file, i.e., MEG, EEG, Demographic, CANTAB results, Genetic Analysis, etc.	String: varchar(40)		

Table 4 data_category specification

Field Name	Description	Data Type	Unit	Note
data_category_id	Data category ID. Primary key.	String: varchar(2)		PK
data_category_name	Category of data file i.e., retrospective, or prospective	String: varchar(40)		

Table 5 data_status specification

Field Name	Description	Data Type	Unit	Note
data_status_id	Data status ID. Primary key.	String: varchar(2)		PK
data_status_name	Status of data file i.e., uploaded, checked, deleted, archived, obsolete or ingested	String: varchar(40)		

4 Data Warehouse Design

In general, the DW is the main repository to store and manage AI-Mind data. This includes the raw data, standardised data, curated data, featured data, trained models, and deployed models. As indicated in Section 1.3 , this document focuses on the first dataset, i.e., the raw data.

There are two types of raw data: the prospective and retrospective data. The prospective data are common in format for all clinical partners; therefore, a common set of database tables of prospective data is sufficient, see Section 4.2. The retrospective data is unique for each clinical partner, therefore separate sets of tables in the database have been prepared for each clinical partner. They are described in Sections 4.3 to 4.5.

The description of the DW file system to handle the raw data files for both the prospective and retrospective data is presented in Section 4.1.

4.1 Data Warehouse File System

The DW file system is organised by data state, data category, and data type, see Figure 5. A Python code will be written to set up the DW file system. Once deployed within TSD, it needs to run only once to prepare all necessary directories.

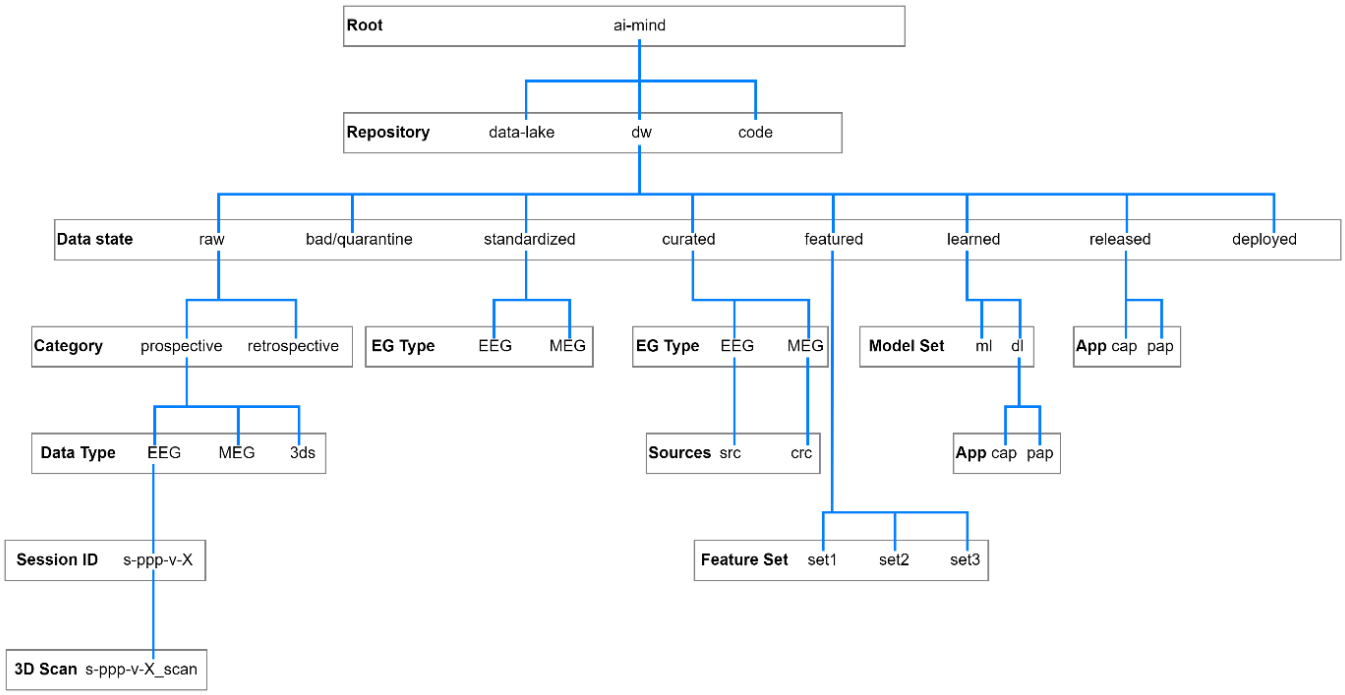

Figure 5 The Data Warehouse File System (adapted from Figure 13 in D4.1).

Given the size of EEG files, they are best suited to be stored in this DW file system.

4.2 Data Warehouse Database Design: Prospective

The DW database design is centralised around the session table (see Figure 6 and Table 8), which contains information about a participant's session, i.e., session ID, participant ID, visit number, and date & time of visit. See Figure 13 for the overall entity relationship diagram for the prospective database.

The session ID is a unique value, which is formed logically using the combination of visit number, participant number, site number and validation letter, all of which are separated by a hyphen ('-'). However, in the DW database, the session ID will be stored as a 5 digit integer (i.e. the validation letter and all hyphens will be removed). Management of the participant ID and the site number is described in Table 6 and Table 7.

A participant's session may have several tasks, for example, cognitive test, EEG recording, blood sample test, socio-demographic, etc, which is managed by task table, see Table 9. The reference of task types is specified in task_type table (Table 10) and status (e.g. raw/loaded, pre-processed etc) is specified in task_status table (Table 11).

Depending on the task type, each task may relate to different tables as follows:

1. **EEG recording**, please refer to ERD in Figure 7 and tables in Section 4.2.2.5. EEG raw files are stored in the DW file system. Only the metadata (information related to the EEG zip file and its content) will be stored in database tables, i.e. eeg_session_context (Table 15), eeg_details (Table 16), and eeg_technical_log (Table 17). The EEG session context is taken from the EEG details and EEG technical log files. Other information related directly to the EEG file (e.g., recording type, recording length, sample rate etc.) will largely be identical across files. As such, this information will not be stored in the DW database. EEG files metadata (Table 12) contains information related to the size, file name and full path. Table 18 stores the 3D scan files.
2. **Cognitive test**, please refer to ERD in Figure 8 and Table 20 in Section 4.2.2.6. The cognitive data consists of a cognitive test variant and the score for each cognitive measure. The cognitive test variant types and the measure types are stored in Table 22 and Table 23, respectively. The cognitive test variants and the scores for each session are stored in Table 21 and Table 24, respectively. The way the cognitive data is stored allows possible change (e.g., by adding or decreasing the variant types and/or measures) for each session of a cognitive test.
3. **Socio-demographic data**, please refer to ERD in Figure 9 and tables in Section 4.2.2.7.
   The main table for socio-demographic data is Table 25. It is assumed that the structure of socio-demographic data will not change throughout the project, however, items of categorical variables/parameters may be added, for example education, employment status, income, housing, ethnicity, and marital status. It is not advised to modify and/or delete any item that was previously stored and used by any record in Table 25. If such kind of action is deemed necessary, then all data records in Table 25 shall be re-evaluated and the value shall be re-assigned with the latest categorical parameter. Categorical variables for socio-demographic data are found in Table 26 to Table 33.
4. **Blood sample test**, tbd.
5. **MEG recording,** MEG raw files are stored in the DW file system. Only the metadata (information related to the MEG zip file and its content similarly to EEG metadata) will be stored in database tables (tbd).

4.2.1 Entity Relationship Diagram

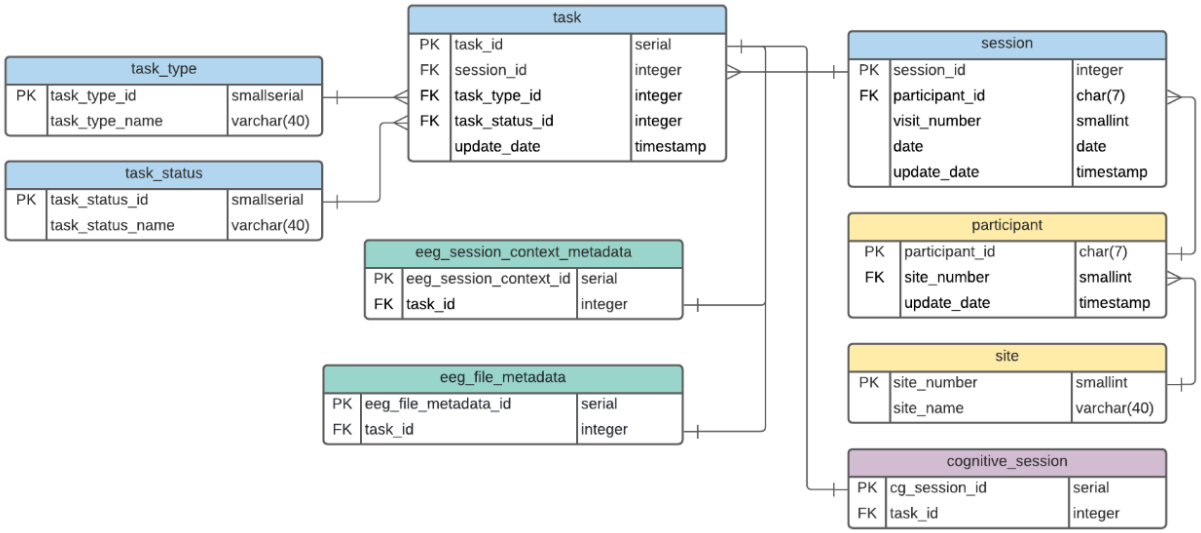

Figure 6 ERD for session and task.

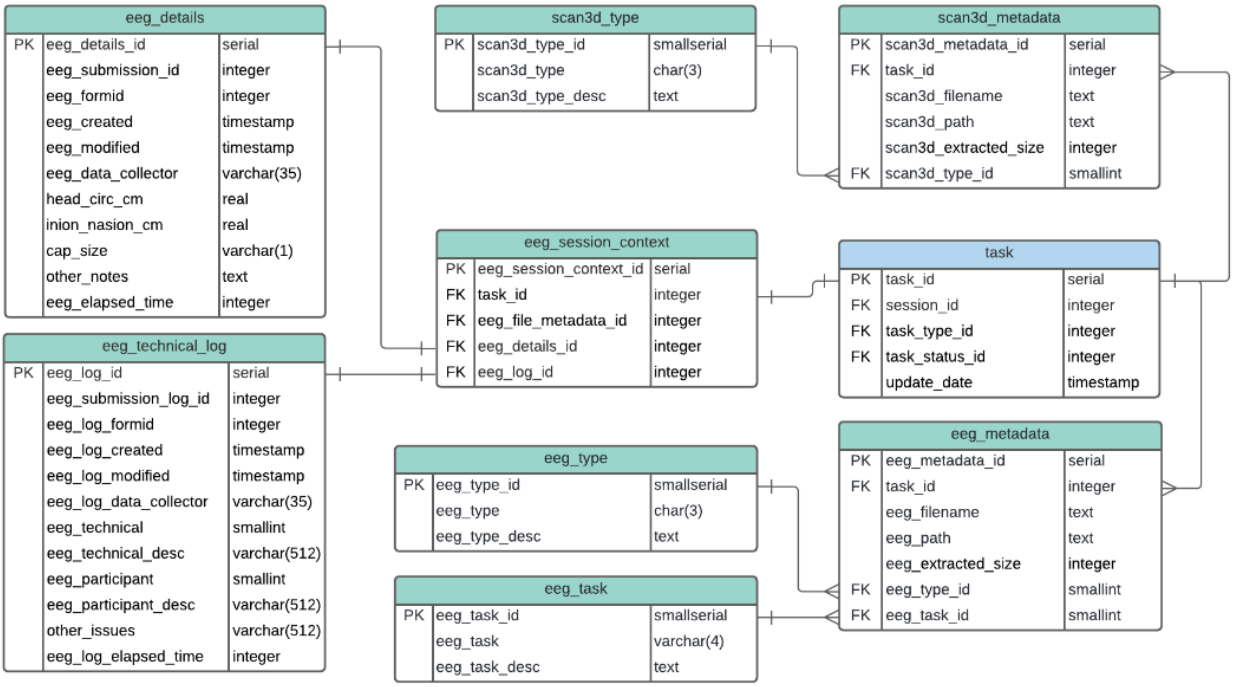

Figure 7 ERD for EEG related data.

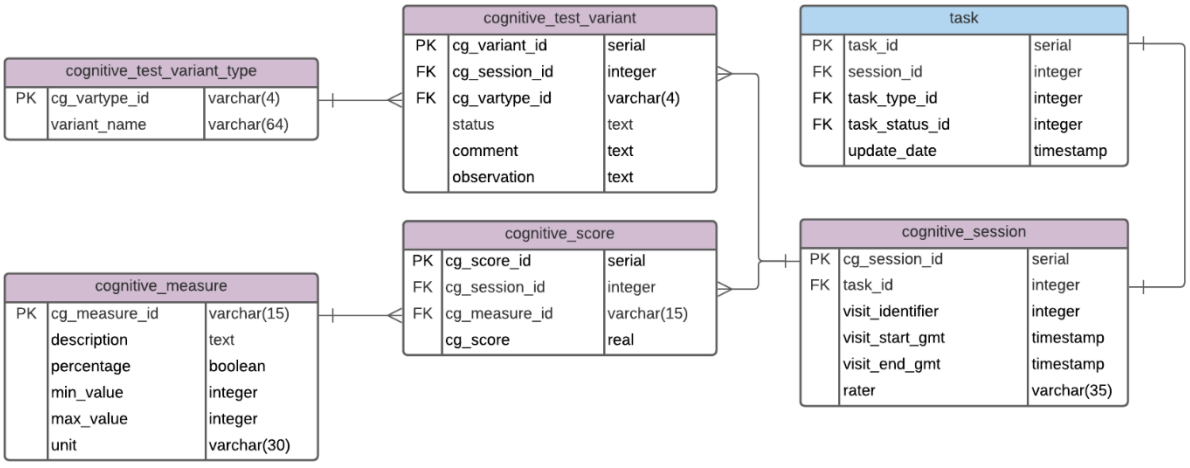

Figure 8 ERD for cognitive related data (CANTAB).

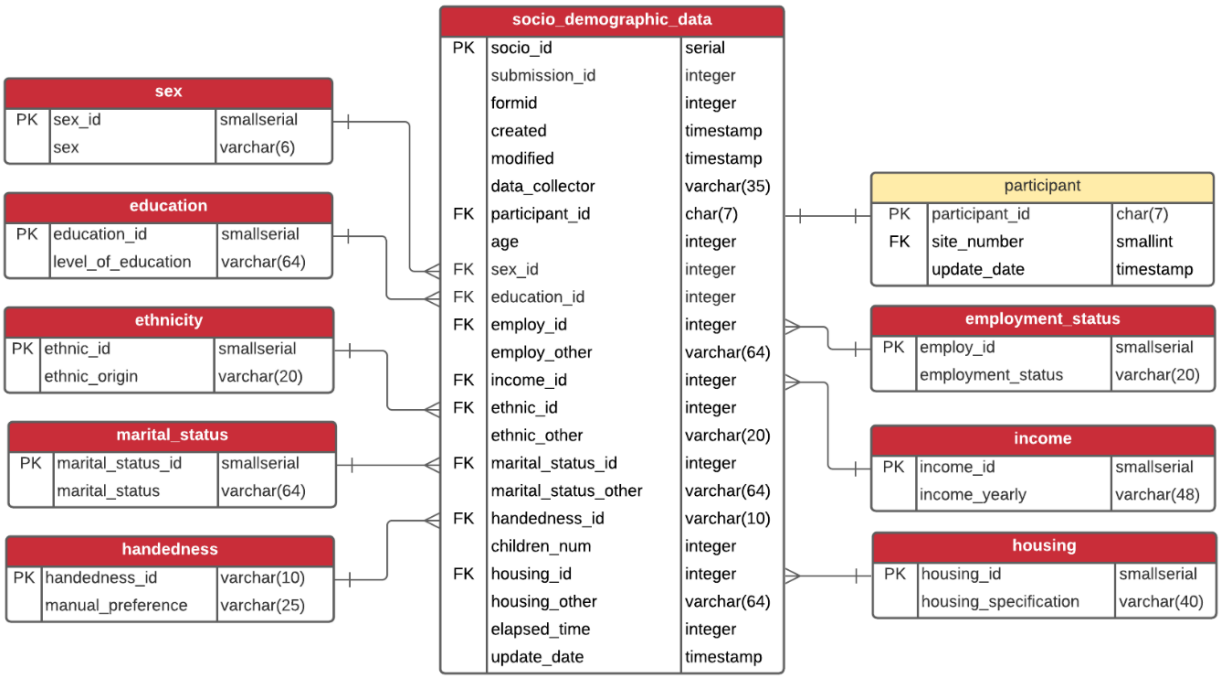

Figure 9 ERD for socio-demographic related data

4.2.2 Tables and Data Fields Specification – Prospective Data

4.2.2.1 Participant Data

Table 6 participant data specification

Field Name	Description	Data Type	Unit	Note
participant_id	A unique ID assigned to the participant on their first registration, which consists of site number (1 digit), participant number (3 digits), and control letter (1 letter) connected with hyphens ('-'). Primary key.	Fixed-length string of 7 characters: char(7)		PK
site_number	Site ID the participant belongs to.	Integer: smallint		FK, link to table site
update_date	The visit time & date of the participant.	Date and time: timestamp		

4.2.2.2 Site Data

Table 7 site specification

Field Name	Description	Data Type	Unit	Note
site_number	Site ID. Primary key.	Integer: smallint		PK
site_name	Name of the site/city the session is conducted.	String: varchar(40)		

4.2.2.3 Session Data

Table 8 session specification

Field Name	Description	Data Type	Unit	Note
session_id	Unique value of which a combination of visit number, participant number and site number. The hyphens and control letter are removed. Primary key.	Integer (5 digits): integer		PK
participant_id	A unique ID assigned to the participant on their first registration, which consists of site number (1 digit), participant number (3 digits), and control letter (1 letter) connected with hyphens (‘-‘).	Fixed-length string of 7 characters: char(7)		FK, link to table participant
visit_number	The visit number of the participant.	Integer (1 digit): smallint		
date	Date and time of visit .	Date and time: timestamp		
update_date	Updated date and time of visit in case of modification of the document/other issues .	Date and time: timestamp		

4.2.2.4 Task Data

Table 9 task specification

Field Name	Description	Data Type	Unit	Note
task_id	A unique number assigned to the task (cognitive test, bio-sample test, EEG recording, MEG recording) during a particular session. Primary key.	Auto increment Integer: serial		PK
session_id	Unique value of which a combination of visit number, participant number and site number. The hyphens and control letter are removed.	Integer (5 digits): integer		FK, link to table session
task_type_id	The ID of task type that will be conducted, i.e., cognitive analysis, bio-sample analysis, EEG recording, MEG recording.	Integer		FK, link to table task_type
task_status_id	The ID of the status of the task .	Integer		FK, link to table task_status
update_date	The time & date when the task was initiated.	Timestamp		

Table 10 task_type specification

Field Name	Description	Data Type	Unit	Note

task_type_id	Task type ID. Primary key.	Auto increment Integer: smallserial		PK
task_type_name	The type of task that will be conducted, i.e., cognitive analysis, bio-sample analysis, EEG recording, MEG recording	String: varchar(40)		

Table 11 task_status specification

Field Name	Description	Data Type	Unit	Note
task_status_id	Task status ID, primary key	Auto increment Integer: smallserial		PK
task_status_name	The status of the task i.e., Raw/loaded, pre-processed, QA Checked, Deleted, etc	String: varchar(40)		

4.2.2.5 EEG Metadata

Table 12 eeg_metadata specification

Field Name	Description	Data Type	Unit	Note
eeg_metadata_id	EEG metadata ID. Primary key.	Auto increment integer: serial		PK
task_id	Task ID that is created for EEG recording.	Integer		FK, link to table task
eeg_filename	EEG filename.	Integer		
eeg_path	EEG filepath, without filename.	Integer		
eeg_extracted_size	EEG filesize.	Integer	KB	
eeg_type_id	EEG file type.	Categorical: integer		FK, link to table eeg_type
eeg_task_id	EEG task ID.	Categorical: integer		FK, link to table eeg_task

Table 13 eeg_type specification

Field Name	Description	Data Type	Unit	Note
eeg_type_id	EEG type ID. Primary key.	Auto increment integer: smallserial		PK
eeg_type	EEG type: cnt, evt	String: varchar(3)		
eeg_type_desc	The description of EEG file type.	String: text		

Table 14 eeg_task specification

Field Name	Description	Data Type	Unit	Note
eeg_task_id	EEG task ID. Primary key.	Auto increment integer: smallserial		PK
eeg_task	EEG task: AR, EO-1, EC-1, EO-2, EC-2	String: varchar(4)		

eeg_task_desc	The description of EEG task.	String: text		

Table 15 eeg_session_context specification

Field Name	Description	Data Type	Unit	Note
eeg_session_context_id	EEG session context metadata ID. Primary key.	Auto increment integer: serial		PK
task_id	Task ID that is created for EEG recording.	Integer		FK, link to table task
eeg_file_metadata_id	EEG session context metadata ID. Primary key.	Integer		
eeg_details_id	eeg_details table id.	Integer		FK, link to table eeg_details
eeg_log_id	eeg_technical_log table id.	Integer		FK, link to table eeg_technical_log

Table 16 eeg_details specification

Field Name	Description	Data Type	Unit	Note
eeg_details_id	EEG details id. Primary key.	Auto increment integer: serial		PK
eeg_submission_id	Unique submission number (8 digits).	Integer		
eeg_formid	Unique identifier shared by all submissions to a particular Nettskjema (6 digits).	Integer		
eeg_created	Time for creation of submission.	Date and time: timestamp		
eeg_modified	Not relevant (only applies if allowed to do changes to existing submission).	Date and time: timestamp		
eeg_data_collector	The name (first and last name) of the lab technician/PhD candidate conducting the session/trial.	String: varchar(35)		
head_circ_cm	Circumference of the head of the participant	Float: real	cm	
inion_nasion_cm	Distance between the nasion to the inion of the participant.	Float: real	cm	
cap_size	Cap size used for the participant. Legend: S – Small (47-51 cm); M – Medium (51-56 cm); L – Large (56-61 cm).	Categorical: char(1)		
other_notes	Mention other factors taken into consideration for the EEG recording.	String: text		
eeg_elapsed_time	Total time spent filling in a particular submission.	Integer	seconds	

Table 17 eeg_technical_log specification

Field Name	Description	Data Type	Unit	Note

eeg_log_id	EEG technical log id. Primary key.	Auto increment integer: serial		PK
eeg_submission_log_id	Unique submission number (8 digits).	Integer		
eeg_log_formid	Unique identifier shared by all submissions to a particular Nettskjema (6 digits).	Integer		
eeg_log_created	Time for creation of submission.	Date and time: timestamp		
eeg_log_modified	Not relevant (only applies if allowed to do changes to existing submission).	Date and time: timestamp		
eeg_log_data_collector	The name (first and last name) of the lab technician/PhD candidate conducting the session/trial.	String: varchar(35)		
eeg_technical	Technical issues encountered while acquiring EEG. Legend: 1 - No, 2 – Yes.	Categorical: smallint		
eeg_technical_desc	Description of the technical issue	String: text		
eeg_participant	Participant-related issue encountered during the EEG acquisition. Legend: 1 - No, 2 – Yes.	Categorical: smallint		
eeg_participant_desc	Description of participant-related issue during the EEG acquisition.	String: text		
other_issues	Other issues encountered during the EEG acquisition.	String: text		
eeg_log_elapsed_time	Total time spent filling in a particular submission.	Integer	seconds	

Table 18 scan3d_metadata specification

Field Name	Description	Data Type	Unit	Note
scan3d_metadata_id	3D scan file ID. Primary key.	Auto increment integer: serial		PK
task_id	Task ID that is created for EEG recording.	Integer		FK, link to table task
scan3d_filename	3D scan filename.	Integer		
scan3d_path	3D scan path, without filename.	Integer		
scan3d_extracted_size	3D scan filesize.	Integer	KB	
scan3d_type_id	3D scan file type.	Categorical: integer		FK, link to table scan3d_type

Table 19 scan3d_type specification

Field Name	Description	Data Type	Unit	Note
scan3d_type_id	3D scan file type ID. Primary key.	Auto increment integer: smallserial		PK
scan3d_type	3D scan file type: jpg, mtl, and obj	String: char(3)		

scan3d_type_desc	The description of 3D scan file type: jpg - 3d template the obj file is mapped onto. mtl - a library file with metadata that links the obj to jpg. obj - main file with the data.	String: text		

4.2.2.6 Cognitive Data

Table 20 cognitive_session specification

Field Name	Description	Data Type	Unit	Note
cg_session_id	Cognitive session id. Primary key.	Auto increment integer: serial		PK
task_id	Task id that is generated for cognitive analysis	Integer		FK, link to table task
visit_identifier	Visit number of the participant to the cognitive analysis session	Integer		
visit_start_gmt	Visit start time (in GMT) of the participant in the cognitive analysis session	Date and time: timestamp		
visit_end_gmt	Visit end time (in GMT) of the participant in the cognitive analysis session	Date and time: timestamp		
rater	Full name of the lab technician/PhD candidate conducting the session	String: varchar(35)		

Table 21 cognitive_test_variant specification

Field Name	Description	Data Type	Unit	Note
cg_variant_id	Cognitive variant ID. Primary key	Auto increment integer: serial		PK
cg_session_id	Cognitive session ID	Integer		FK, link to table cognitive_session
cg_vartype_id	Variant type ID	String: varchar(4)		FK, link to table cognitive_test_variant_type
status	Status on variant types during the cognitive analysis session	String: text		
comment	Comments on variant types during the cognitive analysis session	String: text		
observation	Observations on variant types during the cognitive analysis session	String: text		

Table 22 cognitive_test_variant_type specification

Field Name	Description	Data Type	Unit	Note
cg_vartype_id	Cognitive variant type ID, primary key	String of 4 characters: varchar(4)		PK
variant_name	Names of the different types of cognitive variants	String: varchar(64)		

Table 23 cognitive_measure specification

Field Name	Description	Data Type	Unit	Note
cg_measure_id	Cognitive measure ID. Primary key.	String of 15 characters: varchar(15)		PK
description	Description of the cognitive measure	String: text		
percentage	To check if the provided data is a percentage or not	Boolean		
min_value	Minimum allowable value for the data	Integer		
max_value	Maximum allowable value for the data	Integer		
unit	unit of measure of the data	String: varchar(30)		

Table 24 cognitive_score specification

Field Name	Description	Data Type	Unit	Note
cg_score_id	Cognitive score ID. Primary key.	Auto increment integer: serial		PK
cg_session_id	Cognitive session ID	Integer		FK, link to table cognitive_session
cg_measure_id	Cognitive measure ID	String: varchar(15)		FK, link to table cognitive_measure
cg_score	Quantitative data obtained during the cognitive analysis session	Float: real		

4.2.2.7 Socio-Demographic Data

Table 25 socio_demographic_data specification

Field Name	Description	Data Type	Unit	Note
socio_id	Socio-demographic ID. Primary key	Auto increment integer: serial		PK
submission_id	Unique submission number (8 digits)	Integer		
formid	Unique identifier shared by all submissions to a particular Nettskjema (6 digits)	Integer		
created	Time for creation of submission	Date and time: timestamp		

modified	Not relevant (only applies if allowed to do changes to existing submission)	Date and time: timestamp		
data_collector	The name (first and last name) of the lab technician/PhD candidate conducting the session/trial	String: varchar(35)		
participant_id	A unique ID assigned to the participant on their first registration, which consists of site number (1 digit), participant number (3 digits), and control letter (1 letter) connected with hyphens ('-').	Fixed-length string of 7 characters: char(7)		FK, link to table participant
age	Age of the participant	Integer		
sex_id	Sex ID of the participant	Categorical, integer		FK, link to table sex
education_id	Level of education ID of the participant.	Categorical, integer		FK, link to table education
marital_status_id	Marital status ID of the participant.	Categorical, integer		FK, link to table marital_status
marital_status_other	Description of other marital status	String: varchar(64)		
handedness_id	Preferred handedness ID of the participant.	Categorical, string: varchar(10)		FK, link to table handedness
ethnic_id	Ethnicity ID of the participant.	Categorical, integer		FK, link to table ethnicity
ethnic_other	Description of other ethnic origin	String: varchar(20)		
employ_id	Employment status ID of the participant.	Categorical, integer		FK, link to table employment_status
employ_other	Description of other employment status	String: varchar(64)		
income_id	Yearly income ID of the participant.	Categorical, integer		FK, link to table income
children_num	Number of children	Integer		
housing_id	Type of housing ID the participant resides in.	Categorical, integer		FK, link to table housing
housing_other	Description of other housing	String: varchar(64)		
elapsed_time	Total time spent filling in a particular submission	Integer	seconds	

update_date	Updated date and time of visit in case of modification of the document/other issues.	Date and time: timestamp		

Table 26 sex specification

Field Name	Description	Data Type	Unit	Note
sex_id	Sex ID. Primary key.	Auto increment integer: smallserial		PK
sex	Sex of the participant. 1 – Male 2 – Female	String: varchar(6)		

Table 27 ethnicity specification

Field Name	Description	Data Type	Unit	Note
ethnic_id	Ethnic ID. Primary key.	Auto increment integer: smallserial		PK
ethnic_origin	Ethnicity of the participant. 1 – White; 2 – Black; 3 – Asian; 4 – Arabic; 5 -Mixed; 6 - Other (please specify).	String: varchar(20)		

Table 28 marital_status specification

Field Name	Description	Data Type	Unit	Note
marital_status_id	Marital status ID. Primary key.	Auto increment integer: smallserial		PK
marital_status	Marital status of the participant. 1 – Single; 2 – Married, civilly united, registered partnership; 3 – Widower; 4 – Separated; 5 – Divorced; 6 – Other (please specify).	String: varchar(64)		

Table 29 handedness specification

Field Name	Description	Data Type	Unit	Note
handedness_id	Handedness ID. Primary key.	String: varchar(10)		PK
manual_preference	Preferred handedness of the participant. right – Right; left – Left; ambidex – Ambidextrous	String: varchar(25)		

Table 30 education specification

Field Name	Description	Data Type	Unit	Note
education_id	Education ID. Primary key.	Auto increment integer: smallserial		PK
level_of_education	Level of education of the participant.	String: varchar(64)		

	1 – Left formal education before age 16 (never started high/upper secondary school); 2 – Left formal education at age 16 (dropped out of high/upper secondary school); 3 – Left formal education at age 17-18 (has not completed any degree); 4 – Undergraduate degree or equivalent; 5 – Master'sdegree or equivalent; 6 – PhD or equivalent			

Table 31 employment_status specification

Field Name	Description	Data Type	Unit	Note
employ_id	Employment status ID. Primary key.	Auto increment integer: smallserial		PK
employment_status	Employment status of the particpant. 1 – Retired; 2 – Full-time employed; 3 – Part-time employed; 4 – Self-employed; 5 – Unemployed; 6 – Other (please specify).	String: varchar(20)		

Table 32 income specification

Field Name	Description	Data Type	Unit	Note
income_id	Income ID. Primary key.	Auto increment integer: smallserial		PK
income_yearly	Annual income of the participant. 1 – Less than €50 000; 2 – More than €50 000 but less than €100 000; 3 – More than €100 000	String: varchar(48)		

Table 33 housing specification

Field Name	Description	Data Type	Unit	Note
housing_id	Housing ID. Primary key.	Auto increment integer: smallserial		PK
housing_specification	Housing description. 1 – Living in a house/an apartment owned by themselves and/or spouse/partner; 2 – Renting a house/an apartment; 3 – No permanent residence; 4 – Other (please specify).	String: varchar(40)		

4.2.2.8 Blood Test Data

tbd.

4.2.2.9 MEG Metadata

Corresponding information to EEG metadata, details tbd.

4.3 Data Warehouse Database Design: Retrospective - OUS

The retrospective-OUS data consists of a tsv file, which contains the participant sex and age, and a set of zip files, which each contain the EEG file and other related files (Figure 10, Section 4.3.1). The metadata of the OUS retrospective data is stored and managed in the database tables as described in Table 34 and Table 35 (Section 4.3.2).

4.3.1 Entity Relationship Diagram

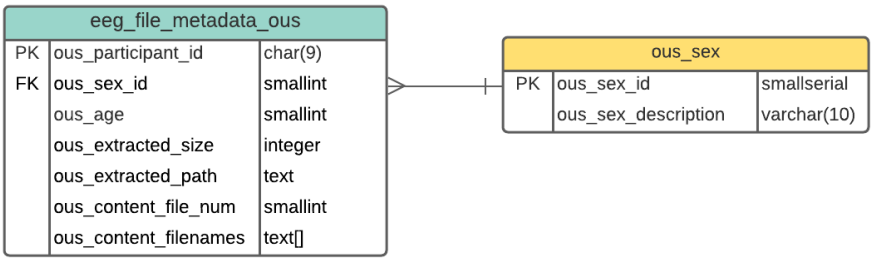

Figure 10 ERD for retrospective data for OUS.

4.3.2 Tables and Data Fields Specification – Retrospective Data: OUS

Table 34 eeg_file_metadata_ous specification

Field Name	Description	Data Type	Unit	Note
ous_participant_id	Participant ID. Primary key: sub-00000	String of 9 characters: char(9)		PK
ous_sex_id	Sex ID.	Integer: smallint		FK, link to table ous_sex
ous_age	Age of the participant	Integer: smallint		
ous_extracted_size	Extracted size of the files from the zip file	Integer	KB	
ous_extracted_path	Path name where the extracted files has been stored	String: text		
ous_content_ file_num	Number of extracted files.	Integer: smallint		
ous_content_filenames	Name of extracted files.	Array of string: text[]		

Table 35 ous_sex specification

Field Name	Description	Data Type	Unit	Note
ous_sex_id	Sex ID, primary key	Auto increment integer: smallserial		PK
ous_sex_description	Sex of the participant	String: varchar(10)		

4.4 Data Warehouse Database Design: Retrospective – UCM

The retrospective-UCM data consists of an excel file, which contains the participant sex and age, diagnosis, and MMSE result (Figure 11, Section 4.4.1). For each participant, the MEG file is zipped. The metadata of the UCM retrospective data is stored and managed in the database tables as described in Table 36, Table 37 and Table 38 (Section 4.4.2).

4.4.1 Entity Relationship Diagram

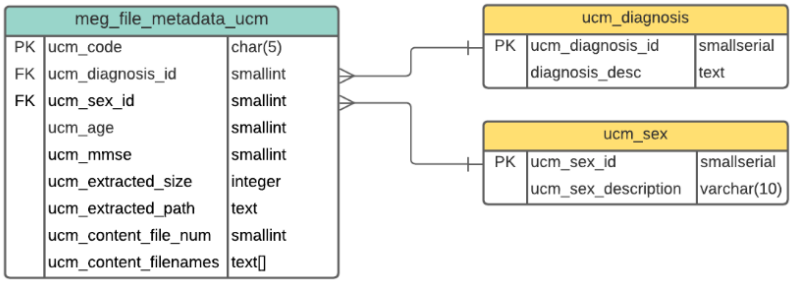

Figure 11 ERD for retrospective data for UCM.

4.4.2 Tables and Data Fields Specification – Retrospective Data: UCM

Table 36 meg_file_metadata_ucm specification

Field Name	Description	Data Type	Unit	Note
ucm_code	Code for the participant. Primary Key	String: char(5)		PK
ucm_diagnosis_id	Diagnosis ID. Legend: 0 – Control; 1 - MCI	Integer: smallint		FK, link to table ucm_diagnosis
ucm_sex_id	Sex ID. Legend: 1 – Male; 2 - Female	Integer: smallint		FK, link to table ucm_sex
ucm_age	Age of the participant	Integer: smallint		
ucm_mmse	MMSE score of the participant. Range: 0 - 30	Integer: smallint		
ucm_extracted_size	Extracted size of the files from the zip file	Integer	KB	
ucm_extracted_path	Path name where the extracted files has been stored	String: text		
ucm_content_ file_num	Number of files in the zip file	Integer: smallint		
ucm_content_filenames	Name of files in the zip file	Array of string: text[]		

Table 37 ucm_sex specification

Field Name	Description	Data Type	Unit	Note
ucm_sex_id	Sex ID, primary key. Legend: 1 – Male; 2 - Female	Auto increment integer: smallserial		PK
ucm_sex_description	Sex of the participant	String: varchar(10)		

Table 38 ucm_diagnosis specification

Field Name	Description	Data Type	Unit	Note
ucm_diagnosis_id	Diagnosis ID, primary key. Legend: 0 – Control; 1 - MCI	Auto increment integer: smallserial		PK
diagnosis_desc	Description of the type of diagnosis	String: text		

4.5 Data Warehouse Database Design: Retrospective - IRCCS/UCSC

The retrospective IRCCS/UCSC data consists of an EEG data zipped file an excel (xls) file for all the subjects. The xls file contains the following data: age, sex, education, MMSE, diagnosis, ApoE value, RAVLT immediate recall, RAVLT delayed recall, Constructional Praxis, Constructional Praxis Landmarks, Rey-Osterreith memory figure copy and immediate recall, Phonetic Verbal Fluency, Semantic Verbal Fluency, Wechsler Adult Intelligence Scale (WAIS-IV) - Digit Span, Raven's Matrices '47, Multiple Features Target Cancellation (MFTC), Naming (SAND), Instrumental activities in daily life (IADL),and The Neuropsychiatric Inventory-Questionnaire (NPI-Q) (Figure 12, Section 4.5.1). The metadata of the IRCCS/UCSC retrospective data is stored and managed in the database tables as described in Table 39, Table 40, Table 41, Table 42 and Table 43 (Section 4.5.2).

4.5.1 Entity Relationship Diagram

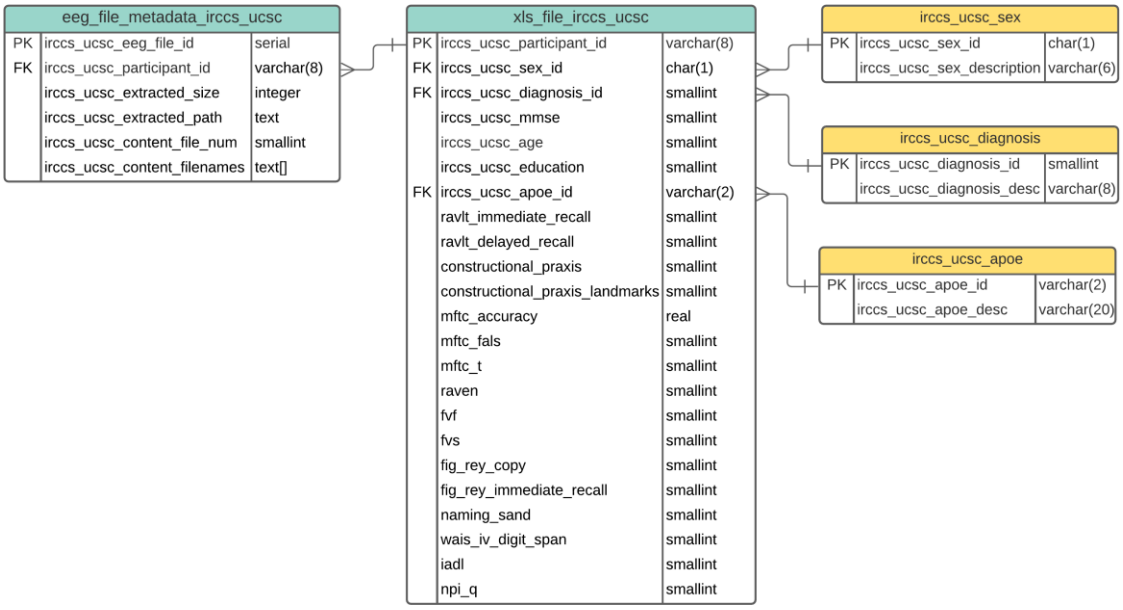

Figure 12 ERD for retrospective data for IRCCS/UCSC

4.5.2 Tables and Data Fields Specification – Retrospective Data: IRCCS/UCSC

Table 39 eeg_file_metadata_irccs_ucsc specification

Field Name	Description	Data Type	Unit	Note
irccs_ucsc _eeg_file_id	EEG file ID. Primary key.	Auto increment integer: serial		PK
irccs_ucsc _participant_id	Participant ID assigned to the participant	String: varchar(8)		FK, link to table xls_file_irccs_ucsc
irccs_ucsc _extracted_size	Extracted size of the files from the zip file	Integer	KB	
irccs_ucsc _extracted_path	Path name where the extracted files has been stored	String: text		
irccs_ucsc _content_ file_num	Number of extracted files	Integer: smallint		
irccs_ucsc _content_filenames	Name of extracted files	Array of string: text[]		

Table 40 xls_file_irccs_ucsc specification

Field Name	Description	Data Type	Unit	Note
irccs_ucsc _participant_id	Code for the participant. Primary key.	String: varchar(8)		PK
irccs_ucsc _sex_id	Sex ID. Legend: M – Male; F – Female	Categorical string: char(1)		FK, link to table irccs_uscs_sex

irccs_ucsc _diagnosis_id	Diagnosis ID. Legend: 0 – no MCI; 1 – has MCI	Integer: smallint		FK, link to table irccs_uscs_diagnosis
irccs_ucsc _mmse	MMSE score of the participant. Range: 0 - 30	Integer: smallint		
irccs_ucsc _age	Age of the participant	Integer: smallint		
irccs_ucsc _education	Number of years of education	Integer: smallint		
irccs_ucsc _apoe_id	APOE ID. Legend: C = Carriers ; NC= Non-Carriers for the epsylon-4 alele (the risk alele of the APOE gen).	Categorical string: char(2)		FK, link to table irccs_uscs_apoe
ravlt_immediate_recall	Number of words recalled immediately. Range: 0 - 75	Integer: smallint		
ravlt_delayed_recall	Number of words recalled after 20 minutes delay. Range: 0 - 15	Integer: smallint		
constructional_praxis	Copy of geometrical drawings. Range: 0 – 12	Integer: smallint		
constructional_praxis_landmarks	Copy of geometrical drawings with landmarks. Range: 0 – 70	Integer: smallint		
mftc_accuracy	Multiple Features Target Cancellation (MFTC) accuracy. MFTC is an attentional visual conjunction search test. Range: 0 - 1	Float: real		
mftc_fals	Number of inclusions. Range: 0 - 67	Integer: smallint		
mftc_t	Time of execution Range: 0 – 180 seconds	Integer: smallint	Seconds	
raven	Raven's matrices '47. Range: 0 – 36	Integer: smallint		
fvf	Verbal Fluency Test phonological.	Integer		
fvs	Verbal Fluency Test Semantics	Integer		
fig_rey_copy	Rey-Osterreith memory figure – copy. Range: 0 – 36	Integer: smallint		
fig_rey_immediate_recall	Rey-Osterreith memory figure – immediate recall. Range: 0 – 36	Integer: smallint		
naming_sand	Naming (SAND). Range: 0 – 30	Integer: smallint		
wais_iv_digit_span	Wechsler Adult Intelligence Scale (WAIS-IV) – Digit Span Range: 0 – 9	Integer: smallint		
iadl	IADL ("Instrumental activities in daily life") Range: 0 – 8	Integer: smallint		

npi_q	NPI-Q (The Neuropsychiatric Inventory-Questionnaire). Range: 0 – 144	Integer: smallint		

Table 41 irccs_ucsc_sex specification

Field Name	Description	Data Type	Unit	Note
irccs_ucsc_sex_id	Sex ID, primary key. Legend: M – Male; F – Female	Categorical string: char(1)		PK
irccs_ucsc_sex_description	Sex of the participant	String: varchar(6)		

Table 42 irccs_ucsc_diagnosis specification

Field Name	Description	Data Type	Unit	Note
irccs_ucsc_diagnosis_id	Diagnosis ID, primary key. Legend: 0 – no MCI; 1 – has MCI	Integer: smallint		PK
irccs_ucsc_diagnosis_desc	Description of the type of diagnosis	String: varchar(8)		

Table 43 irccs_ucsc_apoe specification

Field Name	Description	Data Type	Unit	Note
irccs_ucsc_apoe_id	ApoE ID, primary key. Legend: C = Carriers ; NC= Non-Carriers for the epsylon-4 alele (the risk alele of the APOE gen).	String: varchar(2)		PK
irccs_ucsc_apoe_desc	Description of the type of ApoE	String: varchar(20)		

5 Implementation

SQL codes for the DL database and DW database for raw data are ready and presented in Section 7.2. Utilising a database management tool such as pgAdmin or DBeaver, one may run the SQL codes and establish all necessary database tables. TSD has now provided two instances of PostgreSQL database server, i.e., one for testing and another one for production. **The next step is to run the SQL codes and implement the CDR in the TSD Platform.**

Modification to any database table shall be done through the ERD from which the SQL codes will be generated. Accordingly, the database table specification shall be updated as well.

Once all tables are created, we may perform 'create, read, update, delete (CRUD)' operations. Note, the codes for CRUD operations will be developed as part of D2.4.

6 Conclusion

The CDR is responsible for ensuring persistence, consistency, concurrency, integrity, and security across all AI-Mind applications and processes. The detailing in this report of the DL design and the DW design for raw data, will serve as the basis for DW design for the remaining data types (i.e., standardised, curated etc), and the implementation of the CDR in the TSD Platform. This report will be updated as the project evolves and more information is gathered about these remaining data types and their requirements.

7 Appendix

7.1 Entity Relationship Diagram for Data Warehouse Database Model: Prospective

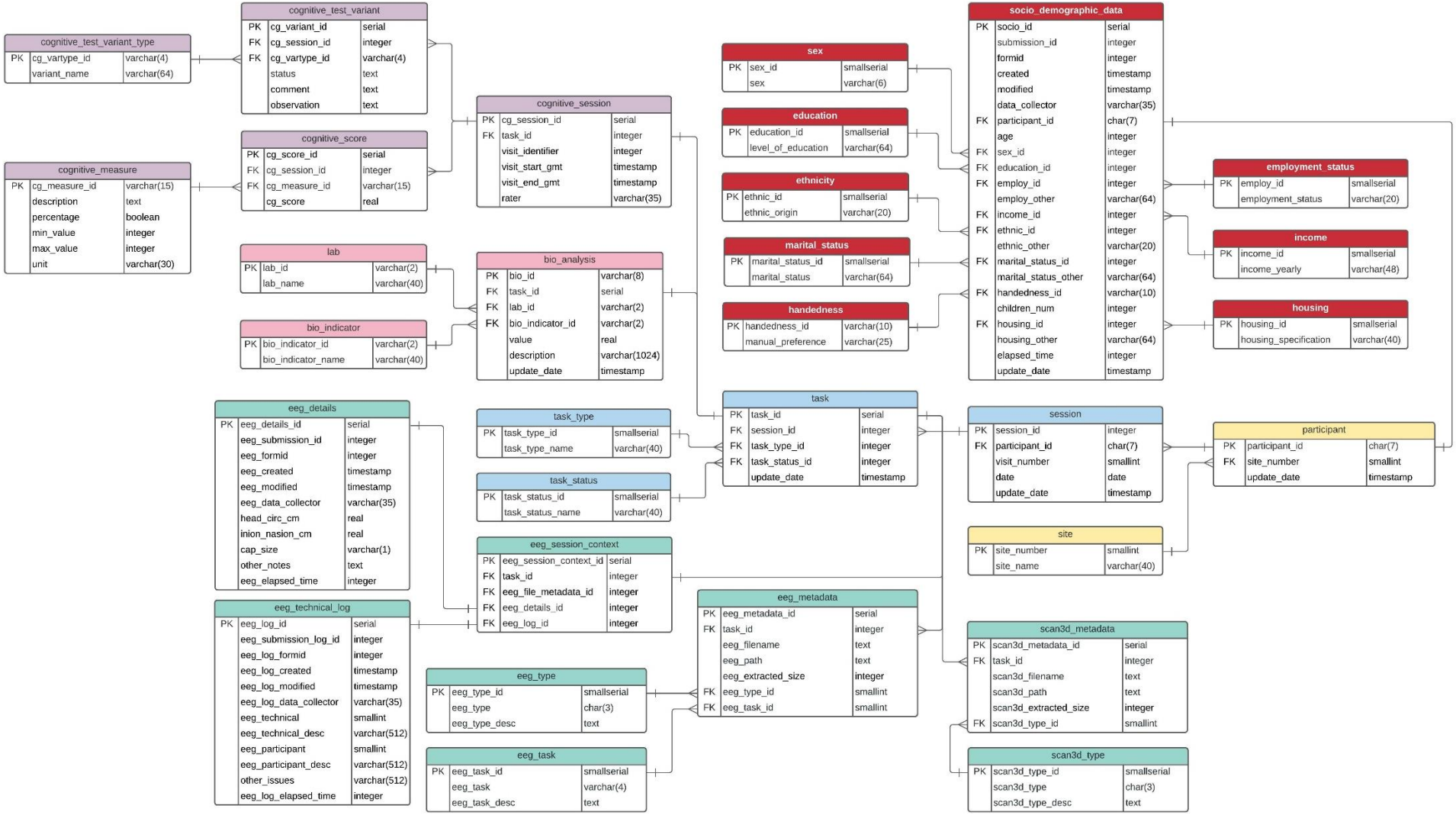

Figure 13 ERD for overall Data Warehouse database model: prospective.

7.2 SQL Codes

7.2.1 SQL Code to Establish Tables related to Data Lake Database

```
CREATE TABLE "data_status" (
  "data_status_id" varchar(2),
  "data_status_name" varchar(40),
  PRIMARY KEY ("data_status_id")
);

CREATE TABLE "data_category" (
  "data_category_id" varchar(2),
  "data_category_name" varchar(40),
  PRIMARY KEY ("data_category_id")
);

CREATE TABLE "source" (
  "source_id" varchar(6),
  "source_name" varchar(40),
  PRIMARY KEY ("source_id")
);

CREATE TABLE "data_type" (
  "data_type_id" varchar(2),
  "data_type_name" varchar(40),
  PRIMARY KEY ("data_type_id")
);

CREATE TABLE "data_catalogue" (
  "dc_id" serial,
  "dc_description" text,
  "source_id" varchar(6),
  "data_type_id" varchar(2),
  "data_category_id" varchar(2),
  "data_status_id" varchar(2),
  "dc_zip_name" varchar(25),
  "dc_zip_size" integer,
  "dc_zip_path" text,
  "dc_content_file_num" smallint,
  "dc_content_filenames" text[],
  "dc_date" timestamp,
  "dc_update_date" timestamp,
  PRIMARY KEY ("dc_id"),
  CONSTRAINT "FK_data_catalogue.data_category_id"
    FOREIGN KEY ("data_category_id")
      REFERENCES "data_category"("data_category_id"),
  CONSTRAINT "FK_data_catalogue.data_type_id"
    FOREIGN KEY ("data_type_id")
      REFERENCES "data_type"("data_type_id"),
  CONSTRAINT "FK_data_catalogue.data_status_id"
    FOREIGN KEY ("data_status_id")
      REFERENCES "data_status"("data_status_id"),
  CONSTRAINT "FK_data_catalogue.source_id"
    FOREIGN KEY ("source_id")
      REFERENCES "source"("source_id")
);
```

7.2.2 SQL code to Establish Tables related to Prospective Data in Data Warehouse

```
CREATE TABLE "cognitive_test_variant_type" (
```

```
  "cg_vartype_id" varchar(4),
  "variant_name" varchar(64),
  PRIMARY KEY ("cg_vartype_id")
);

CREATE TABLE "cognitive_session" (
  "cg_session_id" serial,
  "task_id" integer,
  "visit_identifier" integer,
  "visit_start_gmt" timestamp,
  "visit_end_gmt" timestamp,
  "rater" varchar(35),
  PRIMARY KEY ("cg_session_id")
);

CREATE TABLE "cognitive_test_variant" (
  "cg_variant_id" serial,
  "cg_session_id" integer,
  "cg_vartype_id" varchar(4),
  "status" text,
  "comment" text,
  "observation" text,
  PRIMARY KEY ("cg_variant_id"),
  CONSTRAINT "FK_cognitive_test_variant.cg_vartype_id"
    FOREIGN KEY ("cg_vartype_id")
      REFERENCES "cognitive_test_variant_type"("cg_vartype_id"),
  CONSTRAINT "FK_cognitive_test_variant.cg_session_id"
    FOREIGN KEY ("cg_session_id")
      REFERENCES "cognitive_session"("cg_session_id")
);

CREATE TABLE "eeg_technical_log" (
  "eeg_log_id" serial,
  "eeg_submission_log_id" integer,
  "eeg_log_formid" integer,
  "eeg_log_created" timestamp,
  "eeg_log_modified" timestamp,
  "eeg_log_data_collector" varchar(35),
  "eeg_technical" smallint,
  "eeg_technical_desc" varchar(512),
  "eeg_participant" smallint,
  "eeg_participant_desc" varchar(512),
  "other_issues" varchar(512),
  "eeg_log_elapsed_time" integer,
  PRIMARY KEY ("eeg_log_id")
);

CREATE TABLE "site" (
  "site_number" smallint,
  "site_name" varchar(40),
  PRIMARY KEY ("site_number")
);

CREATE TABLE "income" (
  "income_id" smallserial,
  "income_yearly" varchar(48),
  PRIMARY KEY ("income_id")
);

CREATE TABLE "education" (
  "education_id" smallserial,
  "level_of_education" varchar(64),
  PRIMARY KEY ("education_id")
);

CREATE TABLE "participant" (
  "participant_id" char(7),
  "site_number" smallint,
  "update_date" timestamp,
  PRIMARY KEY ("participant_id"),
```

```
  CONSTRAINT "FK_participant.site_number"
    FOREIGN KEY ("site_number")
      REFERENCES "site"("site_number")
);

CREATE TABLE "session" (
  "session_id" integer,
  "participant_id" char(7),
  "visit_number" smallint,
  "date" date,
  "update_date" timestamp,
  PRIMARY KEY ("session_id"),
  CONSTRAINT "FK_session.participant_id"
    FOREIGN KEY ("participant_id")
      REFERENCES "participant"("participant_id")
);

CREATE TABLE "cognitive_measure" (
  "cg_measure_id" varchar(15),
  "description" text,
  "percentage" boolean,
  "min_value" integer,
  "max_value" integer,
  "unit" varchar(30),
  PRIMARY KEY ("cg_measure_id")
);

CREATE TABLE "task_type" (
  "task_type_id" smallserial,
  "task_type_name" varchar(40),
  PRIMARY KEY ("task_type_id")
);

CREATE TABLE "task_status" (
  "task_status_id" smallserial,
  "task_status_name" varchar(40),
  PRIMARY KEY ("task_status_id")
);

CREATE TABLE "task" (
  "task_id" serial,
  "session_id" integer,
  "task_type_id" integer,
  "task_status_id" integer,
  "update_date" timestamp,
  PRIMARY KEY ("task_id"),
  CONSTRAINT "FK_task.task_type_id"
    FOREIGN KEY ("task_type_id")
      REFERENCES "task_type"("task_type_id"),
  CONSTRAINT "FK_task.session_id"
    FOREIGN KEY ("session_id")
      REFERENCES "session"("session_id"),
  CONSTRAINT "FK_task.task_status_id"
    FOREIGN KEY ("task_status_id")
      REFERENCES "task_status"("task_status_id")
);

CREATE TABLE "lab" (
  "lab_id" varchar(2),
  "lab_name" varchar(40),
  PRIMARY KEY ("lab_id")
);

CREATE TABLE "ethnicity" (
  "ethnic_id" smallserial,
  "ethnic_origin" varchar(20),
  PRIMARY KEY ("ethnic_id")
);

CREATE TABLE "handedness" (
```

```
  "handedness_id" varchar(10),
  "manual_preference" varchar(25),
  PRIMARY KEY ("handedness_id")
);

CREATE TABLE "eeg_details" (
  "eeg_details_id" serial,
  "eeg_submission_id" integer,
  "eeg_formid" integer,
  "eeg_created" timestamp,
  "eeg_modified" timestamp,
  "eeg_data_collector" varchar(35),
  "head_circ_cm" real,
  "inion_nasion_cm" real,
  "cap_size" varchar(1),
  "other_notes" text,
  "eeg_elapsed_time" integer,
  PRIMARY KEY ("eeg_details_id")
);

CREATE TABLE "marital_status" (
  "marital_status_id" smallserial,
  "marital_status" varchar(64),
  PRIMARY KEY ("marital_status_id")
);

CREATE TABLE "employment_status" (
  "employ_id" smallserial,
  "employment_status" varchar(20),
  PRIMARY KEY ("employ_id")
);

CREATE TABLE "sex" (
  "sex_id" smallserial,
  "sex" varchar(6),
  PRIMARY KEY ("sex_id")
);

CREATE TABLE "housing" (
  "housing_id" smallserial,
  "housing_specification" varchar(40),
  PRIMARY KEY ("housing_id")
);

CREATE TABLE "socio_demographic_data" (
  "socio_id" serial,
  "submission_id" integer,
  "formid" integer,
  "created" timestamp,
  "modified" timestamp,
  "data_collector" varchar(35),
  "participant_id" char(7),
  "age" integer,
  "sex_id" integer,
  "education_id" integer,
  "employ_id" integer,
  "employ_other" varchar(64),
  "income_id" integer,
  "ethnic_id" integer,
  "ethnic_other" varchar(20),
  "marital_status_id" integer,
  "marital_status_other" varchar(64),
  "handedness_id" varchar(10),
  "children_num" integer,
  "housing_id" integer,
  "housing_other" varchar(64),
  "elapsed_time" integer,
  "update_date" timestamp,
  PRIMARY KEY ("socio_id"),
  CONSTRAINT "FK_socio_demographic_data.ethnic_id"
```

```
    FOREIGN KEY ("ethnic_id")
      REFERENCES "ethnicity"("ethnic_id"),
  CONSTRAINT "FK_socio_demographic_data.education_id"
    FOREIGN KEY ("education_id")
      REFERENCES "education"("education_id"),
  CONSTRAINT "FK_socio_demographic_data.marital_status_id"
    FOREIGN KEY ("marital_status_id")
      REFERENCES "marital_status"("marital_status_id"),
  CONSTRAINT "FK_socio_demographic_data.income_id"
    FOREIGN KEY ("income_id")
      REFERENCES "income"("income_id"),
  CONSTRAINT "FK_socio_demographic_data.employ_id"
    FOREIGN KEY ("employ_id")
      REFERENCES "employment_status"("employ_id"),
  CONSTRAINT "FK_socio_demographic_data.handedness_id"
    FOREIGN KEY ("handedness_id")
      REFERENCES "handedness"("handedness_id"),
  CONSTRAINT "FK_socio_demographic_data.sex_id"
    FOREIGN KEY ("sex_id")
      REFERENCES "sex"("sex_id"),
  CONSTRAINT "FK_socio_demographic_data.housing_id"
    FOREIGN KEY ("housing_id")
      REFERENCES "housing"("housing_id")
);

CREATE TABLE "bio_indicator" (
  "bio_indicator_id" varchar(2),
  "bio_indicator_name" varchar(40),
  PRIMARY KEY ("bio_indicator_id")
);

CREATE TABLE "cognitive_score" (
  "cg_score_id" serial,
  "cg_session_id" integer,
  "cg_measure_id" varchar(15),
  "cg_score" real,
  PRIMARY KEY ("cg_score_id"),
  CONSTRAINT "FK_cognitive_score.cg_session_id"
    FOREIGN KEY ("cg_session_id")
      REFERENCES "cognitive_session"("cg_session_id"),
  CONSTRAINT "FK_cognitive_score.cg_measure_id"
    FOREIGN KEY ("cg_measure_id")
      REFERENCES "cognitive_measure"("cg_measure_id")
);

CREATE TABLE "bio_analysis" (
  "bio_id" varchar(8),
  "task_id" serial,
  "lab_id" varchar(2),
  "bio_indicator_id" varchar(2),
  "value" real,
  "description" varchar(1024),
  "update_date" timestamp,
  PRIMARY KEY ("bio_id"),
  CONSTRAINT "FK_bio_analysis.bio_indicator_id"
    FOREIGN KEY ("bio_indicator_id")
      REFERENCES "bio_indicator"("bio_indicator_id"),
  CONSTRAINT "FK_bio_analysis.lab_id"
    FOREIGN KEY ("lab_id")
      REFERENCES "lab"("lab_id")
);

CREATE TABLE "eeg_session_context" (
  "eeg_session_context_id" serial,
  "task_id" integer,
  "eeg_file_metadata_id" integer,
  "eeg_details_id" integer,
  "eeg_log_id" integer,
  PRIMARY KEY ("eeg_session_context_id")
);
```

```
CREATE TABLE "scan3d_type" (
  "scan3d_type_id" smallserial,
  "scan3d_type" char(3),
  "scan3d_type_desc" text,
  PRIMARY KEY ("scan3d_type_id")
);

CREATE TABLE "scan3d_metadata" (
  "scan3d_metadata_id" serial,
  "task_id" integer,
  "scan3d_filename" text,
  "scan3d_path" text,
  "scan3d_extracted_size" integer,
  "scan3d_type_id" smallint,
  PRIMARY KEY ("scan3d_metadata_id"),
  CONSTRAINT "FK_scan3d_metadata.scan3d_type_id"
    FOREIGN KEY ("scan3d_type_id")
      REFERENCES "scan3d_type"("scan3d_type_id"),
  CONSTRAINT "FK_scan3d_metadata.task_id"
    FOREIGN KEY ("task_id")
      REFERENCES "task"("task_id")
);

CREATE TABLE "eeg_type" (
  "eeg_type_id" smallserial,
  "eeg_type" char(3),
  "eeg_type_desc" text,
  PRIMARY KEY ("eeg_type_id")
);

CREATE TABLE "eeg_task" (
  "eeg_task_id" smallserial,
  "eeg_task" varchar(4),
  "eeg_task_desc" text,
  PRIMARY KEY ("eeg_task_id")
);

CREATE TABLE "eeg_metadata" (
  "eeg_metadata_id" serial,
  "task_id" integer,
  "eeg_filename" text,
  "eeg_path" text,
  "eeg_extracted_size" integer,
  "eeg_type_id" smallint,
  "eeg_task_id" smallint,
  PRIMARY KEY ("eeg_metadata_id"),
  CONSTRAINT "FK_eeg_metadata.task_id"
    FOREIGN KEY ("task_id")
      REFERENCES "task"("task_id"),
  CONSTRAINT "FK_eeg_metadata.eeg_type_id"
    FOREIGN KEY ("eeg_type_id")
      REFERENCES "eeg_type"("eeg_type_id"),
  CONSTRAINT "FK_eeg_metadata.eeg_task_id"
    FOREIGN KEY ("eeg_task_id")
      REFERENCES "eeg_task"("eeg_task_id")
);
```

7.2.3 SQL Code to Establish Tables related to Retrospective Data : OUS

```
CREATE TABLE "ous_sex" (
  "ous_sex_id" smallserial,
```

```
  "ous_sex_description" varchar(10),
  PRIMARY KEY ("ous_sex_id")
);

CREATE TABLE "eeg_file_metadata_ous" (
  "ous_participant_id" char(9),
  "ous_sex_id" smallint,
  "ous_age" smallint,
  "ous_extracted_size" integer,
  "ous_extracted_path" text,
  "ous_content_file_num" smallint,
  "ous_content_filenames" text[],
  PRIMARY KEY ("ous_participant_id"),
  CONSTRAINT "FK_eeg_file_metadata_ous.ous_sex_id"
    FOREIGN KEY ("ous_sex_id")
      REFERENCES "ous_sex"("ous_sex_id")
);
```

7.2.4 SQL Code to Establish Tables related to Retrospective Data : UCM

```
CREATE TABLE "ucm_sex" (
  "ucm_sex_id" smallserial,
  "ucm_sex_description" varchar(10),
  PRIMARY KEY ("ucm_sex_id")
);

CREATE TABLE "ucm_diagnosis" (
  "ucm_diagnosis_id" smallserial,
  "diagnosis_desc" text,
  PRIMARY KEY ("ucm_diagnosis_id")
);

CREATE TABLE "meg_file_metadata_ucm" (
  "ucm_code" char(5),
  "ucm_diagnosis_id" smallint,
  "ucm_sex_id" smallint,
  "ucm_age" smallint,
  "ucm_mmse" smallint,
  "ucm_extracted_size" integer,
  "ucm_extracted_path" text,
  "ucm_content_file_num" smallint,
  "ucm_content_filenames" text[],
```

```
  PRIMARY KEY ("ucm_code"),
  CONSTRAINT "FK_meg_file_metadata_ucm.ucm_sex_id"
    FOREIGN KEY ("ucm_sex_id")
      REFERENCES "ucm_sex"("ucm_sex_id"),
  CONSTRAINT "FK_meg_file_metadata_ucm.ucm_diagnosis_id"
    FOREIGN KEY ("ucm_diagnosis_id")
      REFERENCES "ucm_diagnosis"("ucm_diagnosis_id")
);
```

7.2.5 SQL Code to Establish Tables related to Retrospective Data : IRCCS/UCSC

```
CREATE TABLE "irccs_ucsc_apoe" (
  "irccs_ucsc_apoe_id" varchar(2),
  "irccs_ucsc_apoe_desc" varchar(20),
  PRIMARY KEY ("irccs_ucsc_apoe_id")
);

CREATE TABLE "irccs_ucsc_sex" (
  "irccs_ucsc_sex_id" char(1),
  "irccs_ucsc_sex_description" varchar(6),
  PRIMARY KEY ("irccs_ucsc_sex_id")
);

CREATE TABLE "irccs_ucsc_diagnosis" (
  "irccs_ucsc_diagnosis_id" smallint,
  "irccs_ucsc_diagnosis_desc" varchar(8),
  PRIMARY KEY ("irccs_ucsc_diagnosis_id")
);
```

```
CREATE TABLE "xls_file_irccs_ucsc" (
  "irccs_ucsc_participant_id" varchar(8),
  "irccs_ucsc_sex_id" char(1),
  "irccs_ucsc_diagnosis_id" smallint,
  "irccs_ucsc_mmse" smallint,
  "irccs_ucsc_age" smallint,
  "irccs_ucsc_education" smallint,
  "irccs_ucsc_apoe_id" varchar(2),
  "ravlt_immediate_recall" smallint,
  "ravlt_delayed_recall" smallint,
  "constructional_praxis" smallint,
  "constructional_praxis_landmarks" smallint,
  "mftc_accuracy" real,
  "mftc_fals" smallint,
  "mftc_t" smallint,
  "raven" smallint,
  "fvf" smallint,
  "fvs" smallint,
  "fig_rey_copy" smallint,
  "fig_rey_immediate_recall" smallint,
  "naming_sand" smallint,
  "wais_iv_digit_span" smallint,
  "iadl" smallint,
  "npi_q" smallint,
  PRIMARY KEY ("irccs_ucsc_participant_id"),
  CONSTRAINT "FK_xls_file_irccs_ucsc.irccs_ucsc_apoe_id"
    FOREIGN KEY ("irccs_ucsc_apoe_id")
      REFERENCES "irccs_ucsc_apoe"("irccs_ucsc_apoe_id"),
  CONSTRAINT "FK_xls_file_irccs_ucsc.irccs_ucsc_sex_id"
    FOREIGN KEY ("irccs_ucsc_sex_id")
      REFERENCES "irccs_ucsc_sex"("irccs_ucsc_sex_id"),
  CONSTRAINT "FK_xls_file_irccs_ucsc.irccs_ucsc_diagnosis_id"
    FOREIGN KEY ("irccs_ucsc_diagnosis_id")
      REFERENCES "irccs_ucsc_diagnosis"("irccs_ucsc_diagnosis_id")
);

CREATE TABLE "eeg_file_metadata_irccs_ucsc" (
  "irccs_ucsc_eeg_file_id" serial,
  "irccs_ucsc_participant_id" varchar(8),
  "irccs_ucsc_extracted_size" integer,
  "irccs_ucsc_extracted_path" text,
  "irccs_ucsc_content_file_num" smallint,
  "irccs_ucsc_content_filenames" text[],
  PRIMARY KEY ("irccs_ucsc_eeg_file_id"),
  CONSTRAINT "FK_eeg_file_metadata_irccs_ucsc.irccs_ucsc_participant_id"
    FOREIGN KEY ("irccs_ucsc_participant_id")
```

```
        REFERENCES "xls_file_irccs_ucsc"("irccs_ucsc_participant_id")
);
```