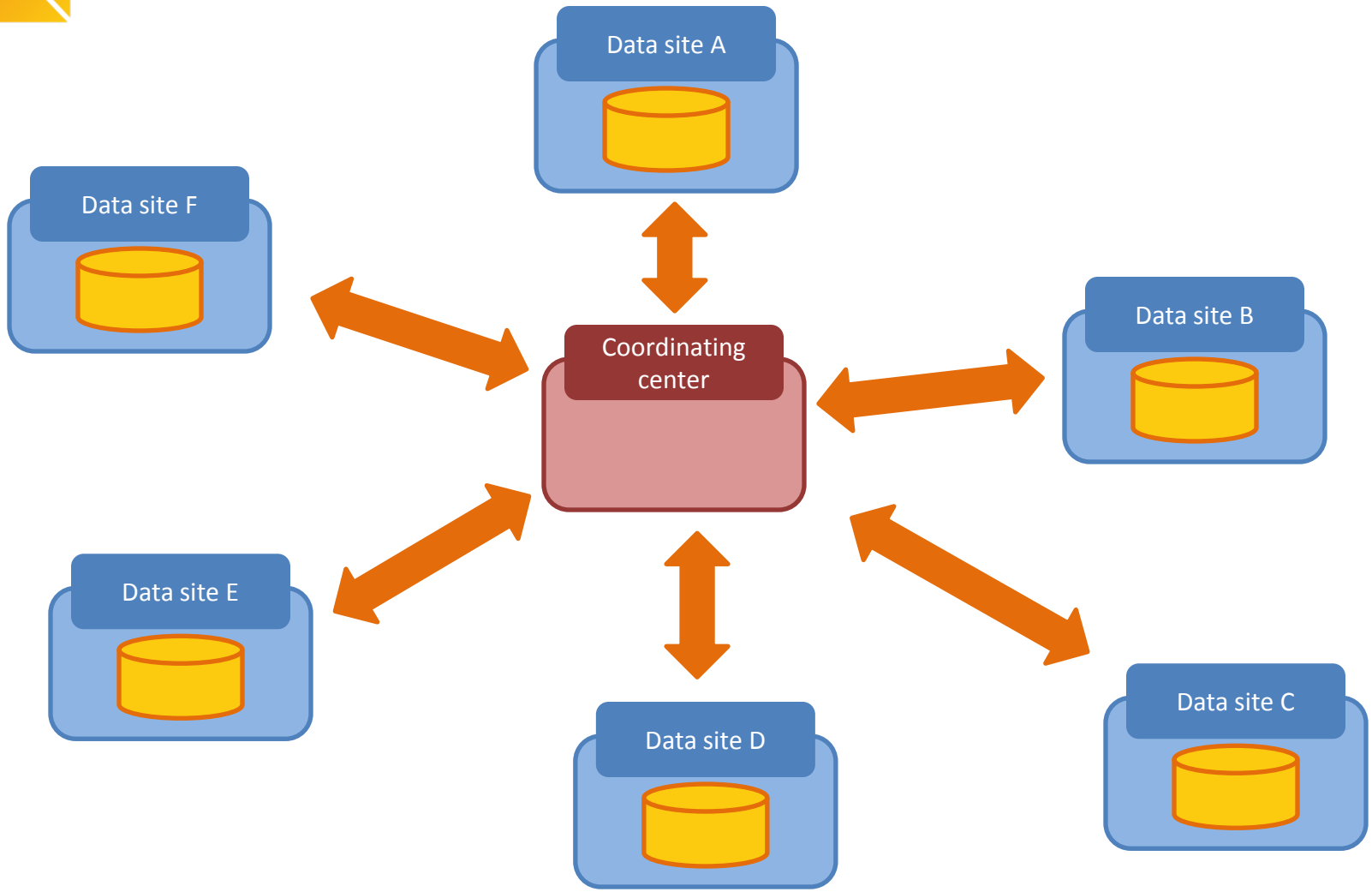
Studies in the OHDSI distributed research network

Martijn Schuemie

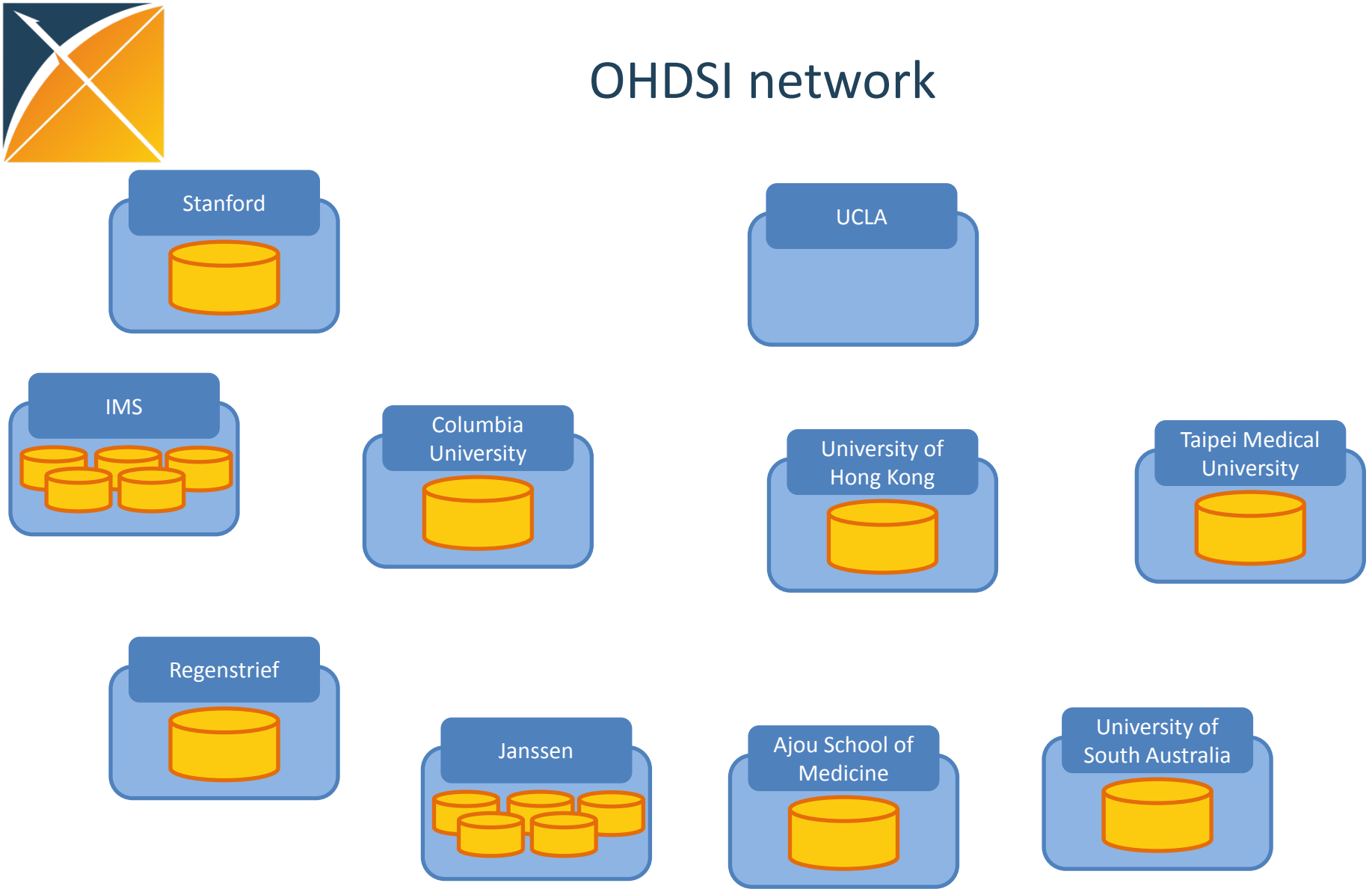
Distributed research network

- Many observational databases in OHDSI
  - large numbers
  - large diversity
- We cannot share patient-level data
- Solution:
  - analysis code ‘visits’ the data
  - only population-level data is shared

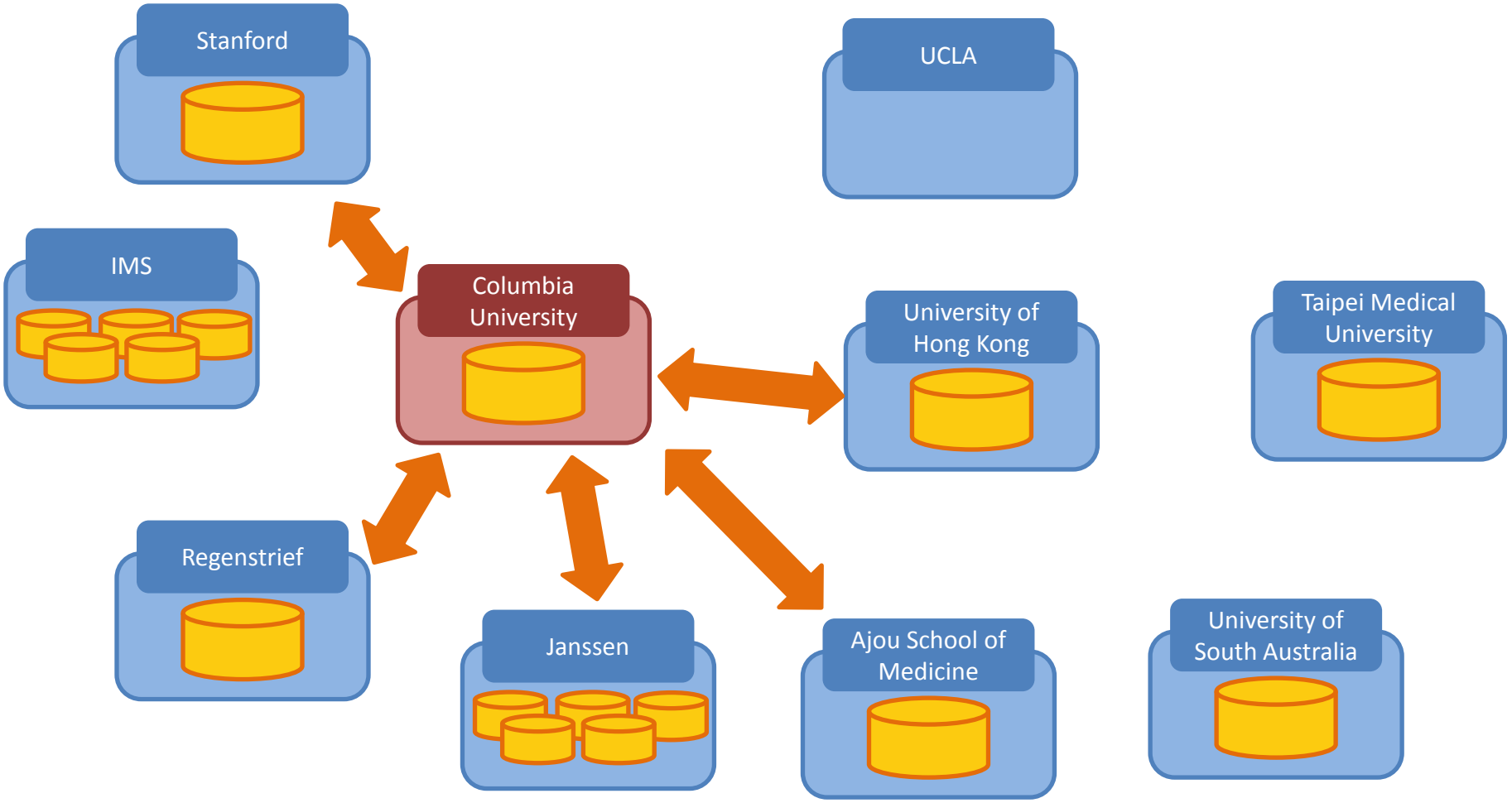
Hub and spoke network



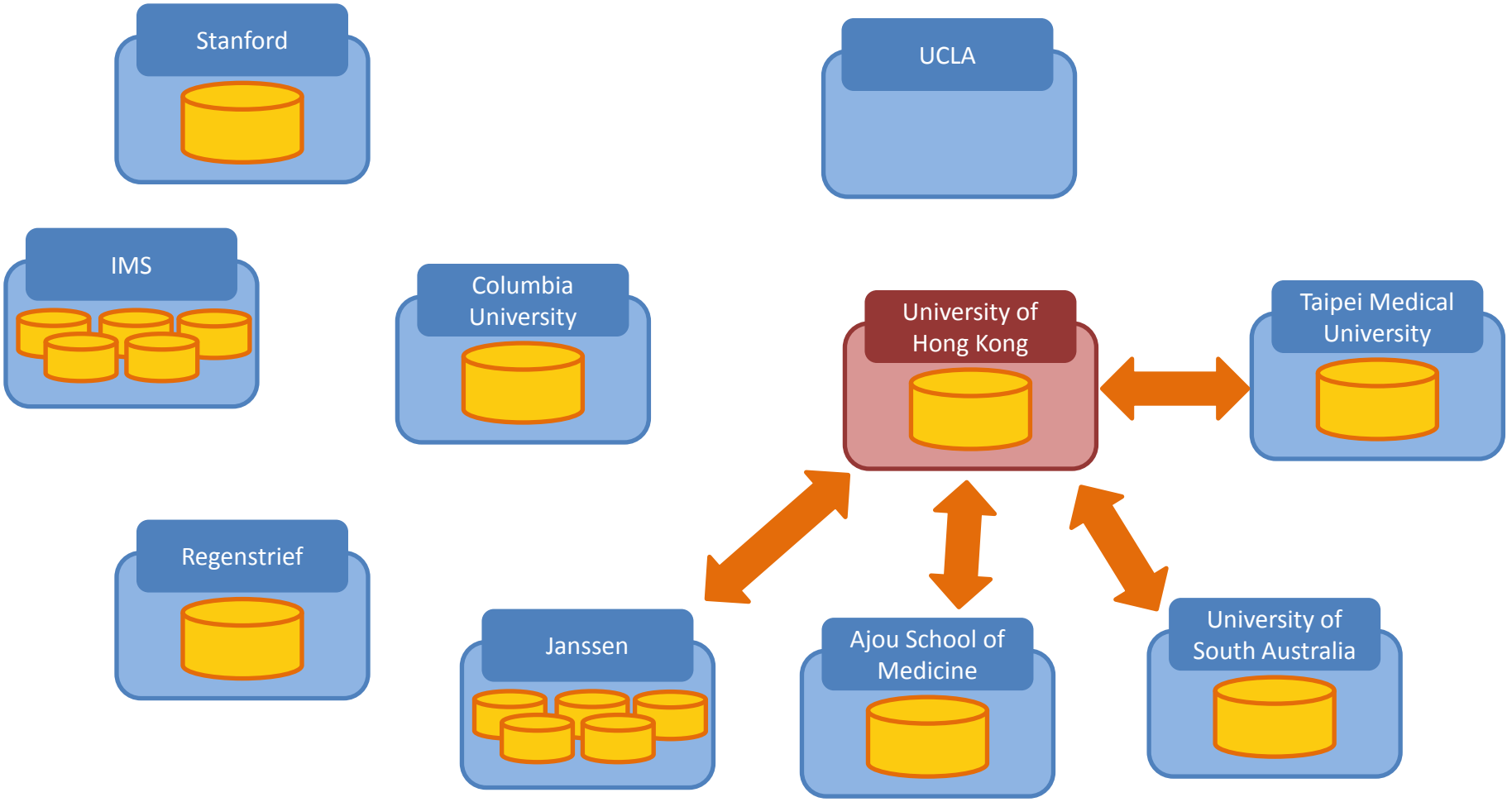
OHDSI network



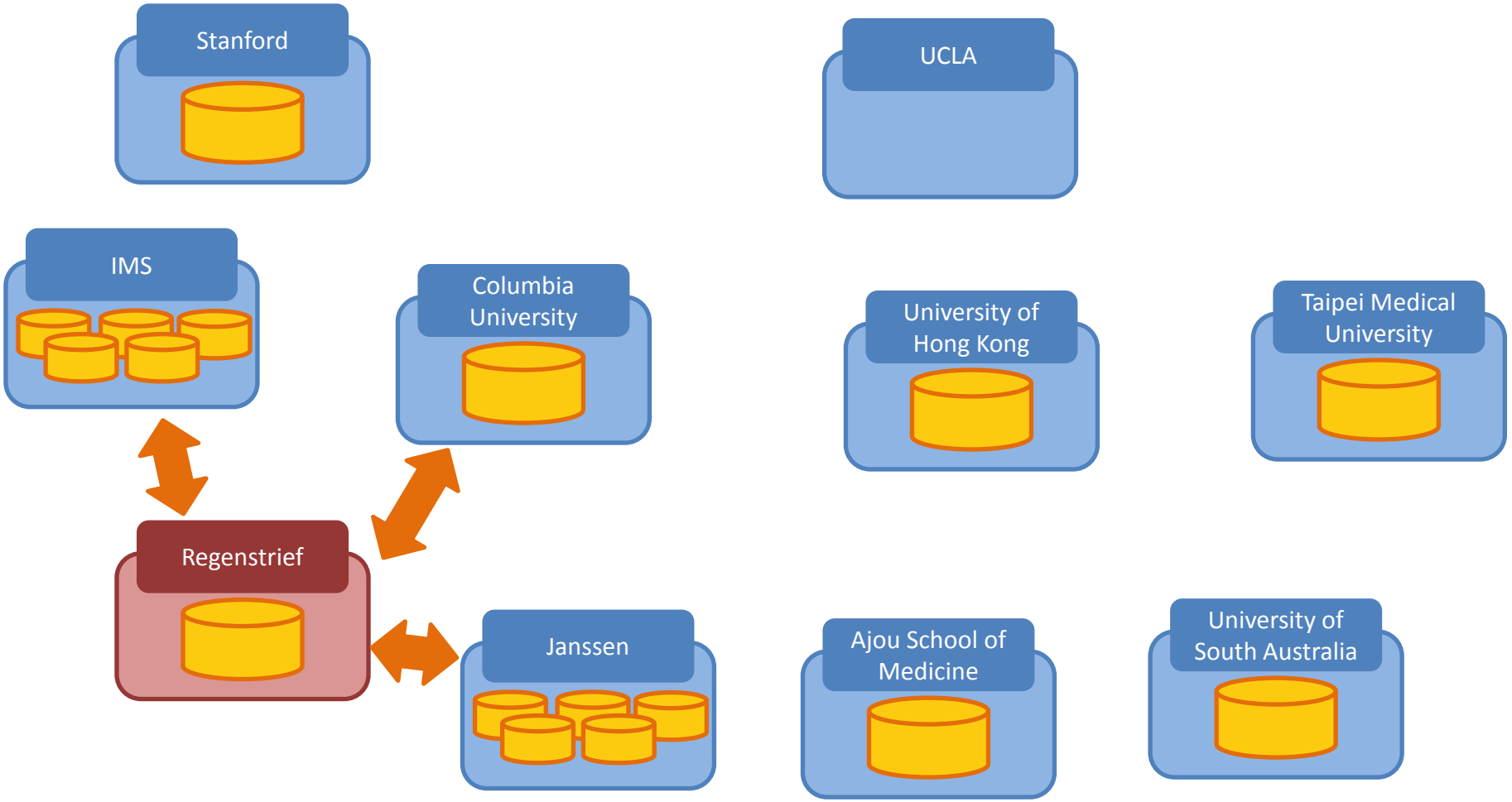
Treatment pathway study



Drug Utilization in Children study



Keppra-angioedema study

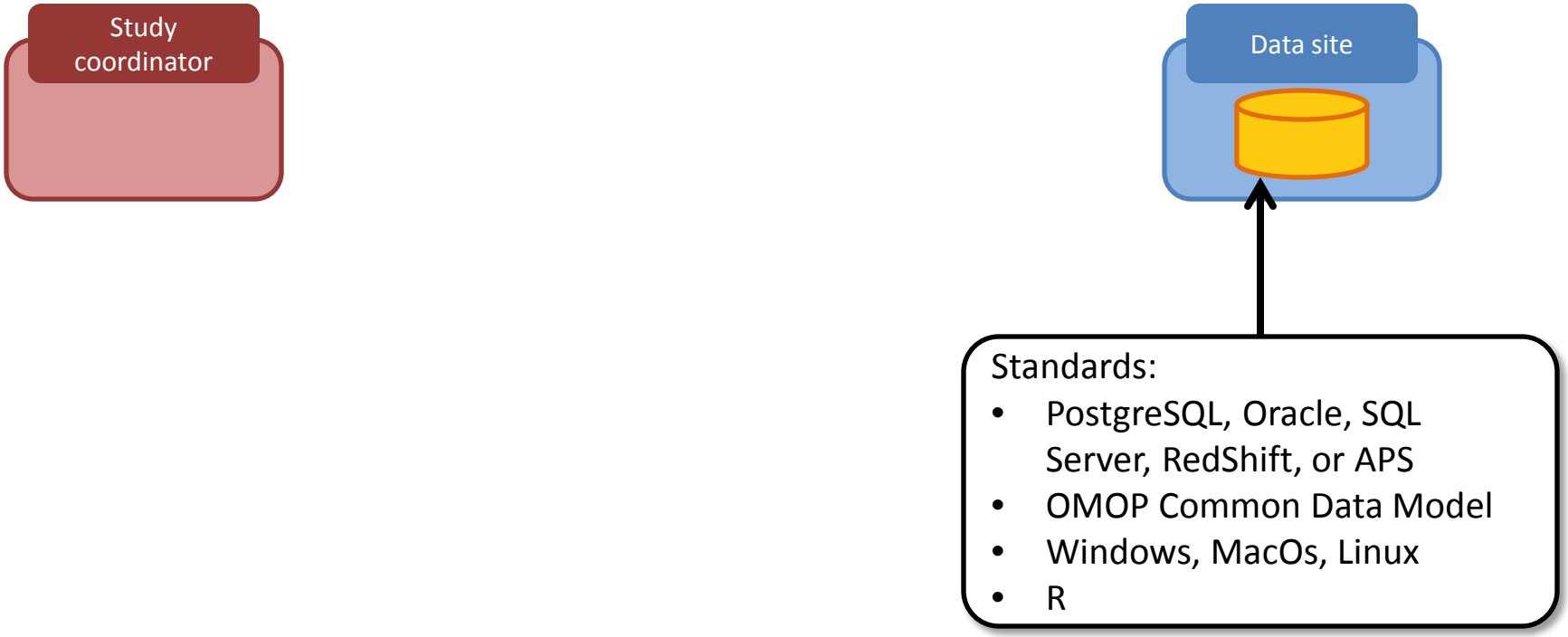


Everyone can initiate and lead a study

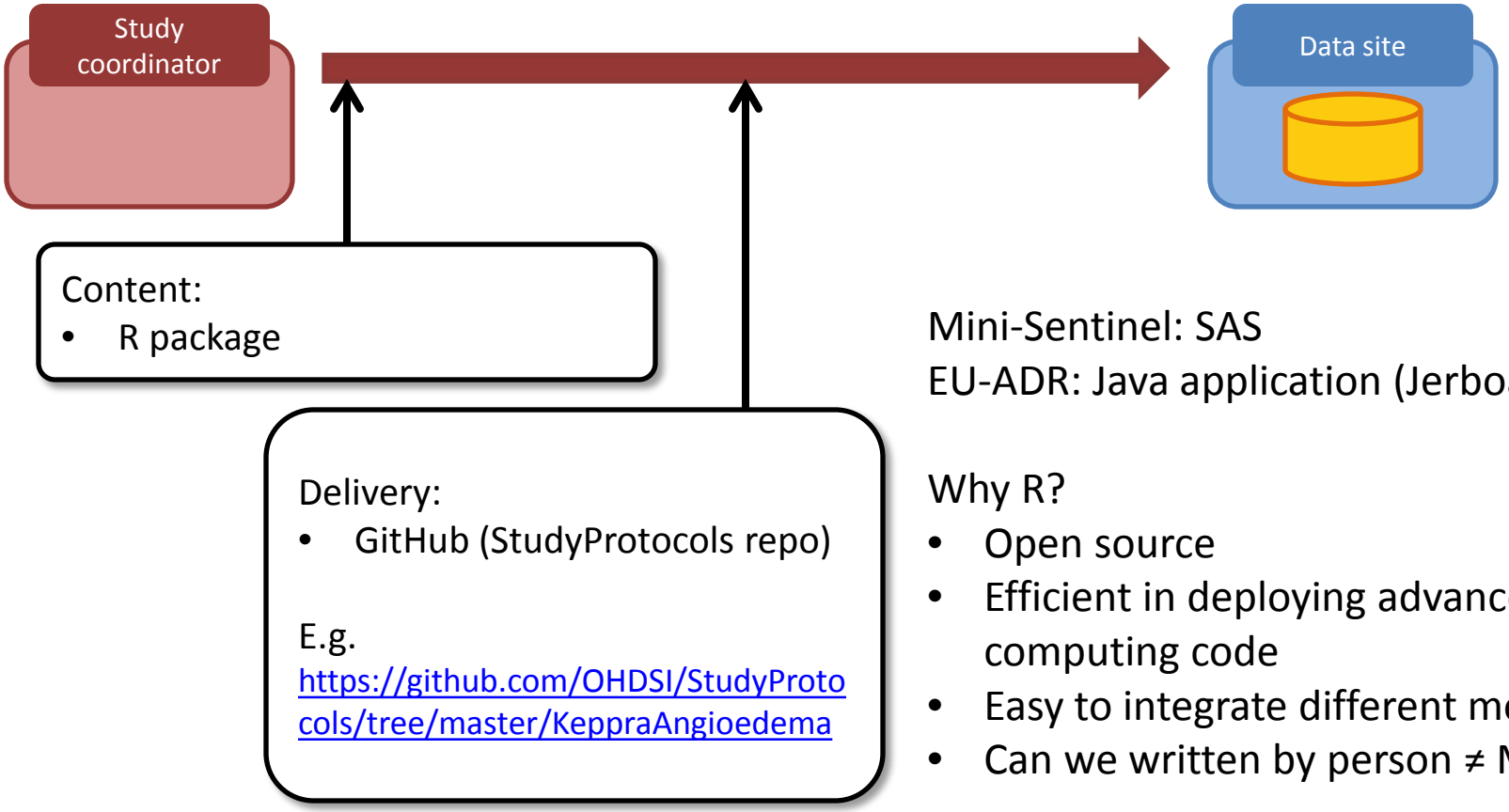
See the Wiki Collaborative Study FAQ:
http://www.ohdsi.org/web/wiki/doku.php?id=research:studies:faq

- Post preliminary protocol on Wiki
- Invite community review
- Post final protocol on Wiki
  - can be used for IRB approval
- Develop study code, post on GitHub
- Test code at at least 2 sites
- Invite sites to join

Implementation



Implementation



Mini-Sentinel: SAS
EU-ADR: Java application (Jerboa)

Why R?

- Open source
- Efficient in deploying advanced computing code
- Easy to integrate different modules
- Can we written by person ≠ Martijn

Implementation

Content: zip file containing
- Plain text
- CSV (comma-separated values)
- PNG (plots)
- …

Needs to be:
- Non-identifiable information
- Human reviewable

Delivery:
- E-mail
- Amazon S3

Study package

- User experience
- Under the hood

Examples: Keppra-angioedema and DrugsInPeds study packages

Study package user experience

Steps for the user:

1. Install the package (and dependencies)
2. Run the analyis
3. Review the results
4. Share the results

Study package user experience

Steps for the user:

1. **Install the package** (and dependencies)
2. Run the analyis
3. Review the results
4. Share the results

Installing the package

Using R's package infrastructure to deploy packages

Not all packages are in CRAN. There's an OHDSI package repository (using "drat")

```
install.packages("drat")
drat::addRepo(c("OHDSI","cloudyr")) # Link to OHDSI packages
install.packages("KeppraAngioedema")
```

Study package user experience

Steps for the user:

1. Install the package (and dependencies)
2. **Run the analyis**
3. Review the results
4. Share the results

Running the analysis

Ideally a single command to run the entire analysis

This should create a folder called ‘export’ with all files that will be shared

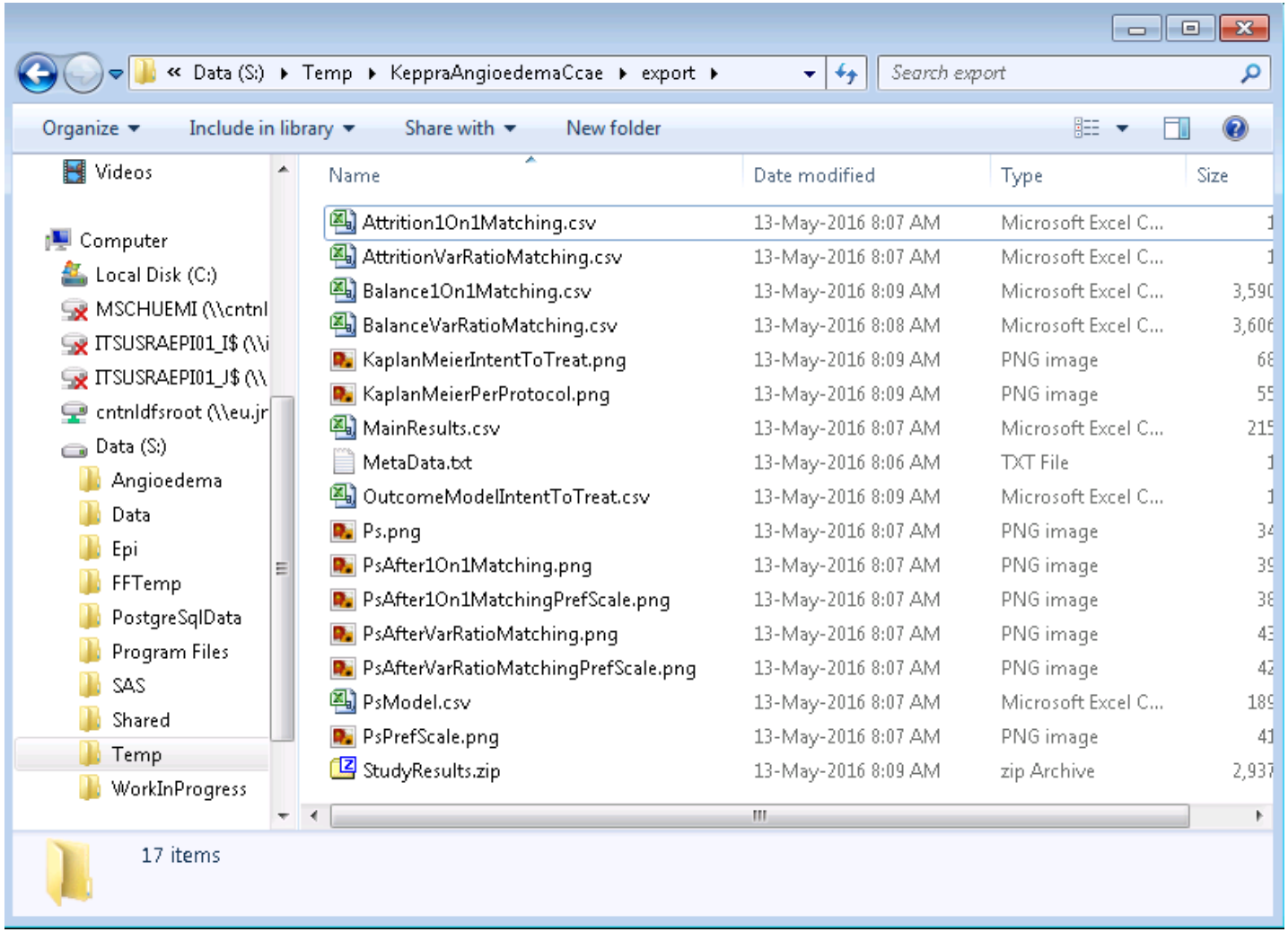
```
library(KeppraAngioedema)

connectionDetails <- createConnectionDetails(dbms = "postgresql",
                                             user = "joe",
                                             password = "secret",
                                             server = "myserver")

execute(connectionDetails,
        cdmDatabaseSchema = "cdm_data",
        workDatabaseSchema = "results",
        studyCohortTable = "ohdsi_keppra_angioedema",
        oracleTempSchema = NULL,
        outputFolder = "c:/temp/study_results",
        maxCores = 4)
```

Study package user experience

Steps for the user:

1. Install the package (and dependencies)
2. Run the analyis
3. **Review the results**
4. Share the results

Review the results



Study package user experience

Steps for the user:

1. Install the package (and dependencies)
2. Run the analyis
3. Review the results
4. **Share the results**

Share the results

Push the StudyResults.zip file to an S3 bucket

```
submitResults("c:/temp/study_results/export", key = "<key>", secret = "<secret>")
```

S3 is Amazon's storage service with secure up and download

One bucket per study

Two accounts per bucket:

- Study coordinator can read and write
- Study participants can write

Contact the coordinating center for your study-specific bucket

Under the hood

Executing the analysis typically entails:

1. Instantiating cohorts
2. Running some SQL against the CDM database
3. Do some further processing in R
4. Write results to the export folder

Instantiating cohorts

Adding CIRCE SQL to the package:

```
OhdsiRTools::insertCirceDefinitionInPackage(2193, "Angioedema")
```

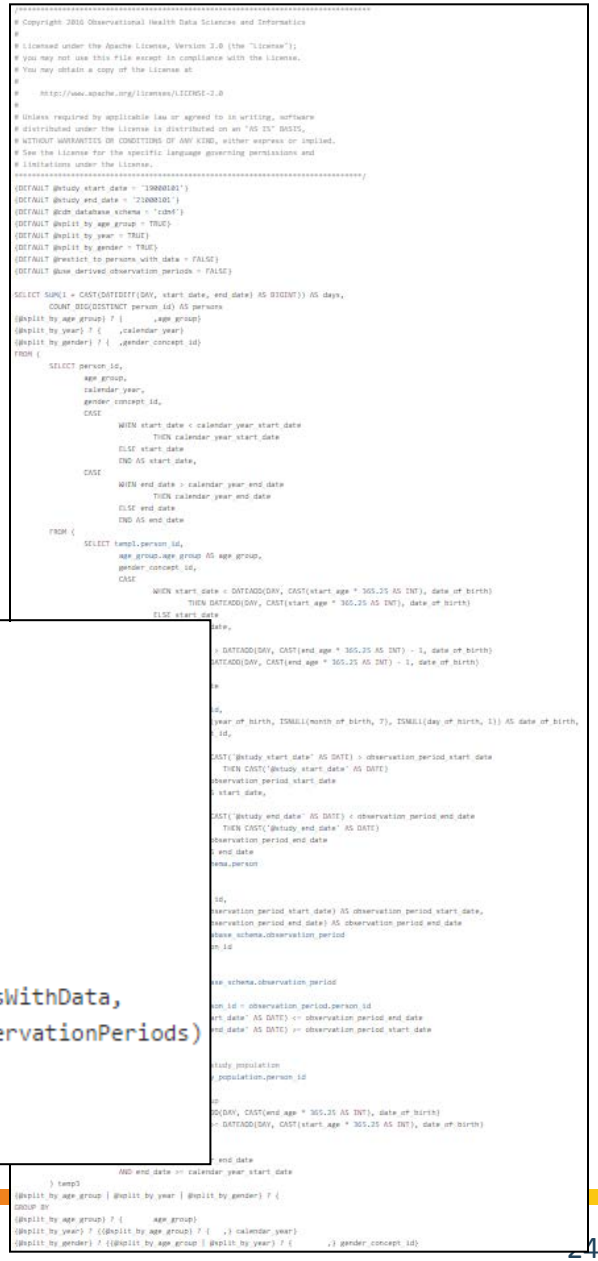
Instantiating the cohort during execution:

```
sql <- SqlRender::loadRenderTranslateSql("Angioedema.sql",
                                         "KeppraAngioedema",
                                         dbms = connectionDetails$dbms,
                                         oracleTempSchema = oracleTempSchema,
                                         cdm_database_schema = cdmDatabaseSchema,
                                         target_database_schema = workDatabaseSchema,
                                         target_cohort_table = studyCohortTable,
                                         cohort_definition_id = 3)
DatabaseConnector::executeSql(conn, sql)
```

We usually do not use the cohort table in the CDM, but create a study-specific cohort table in another schema

Running some SQL against the CDM database

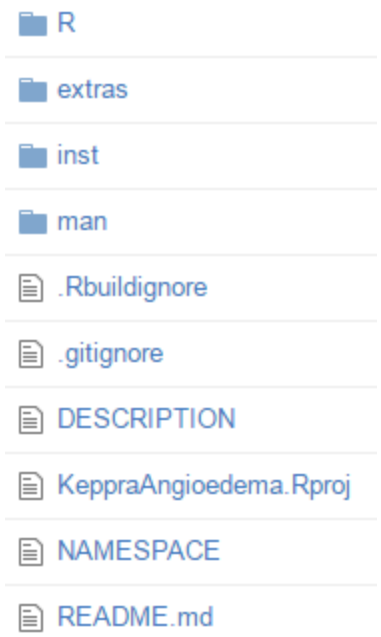
Write OhdsiSql and include in package:



Using SqlRender and DatabaseConnector packages to run in local environment:

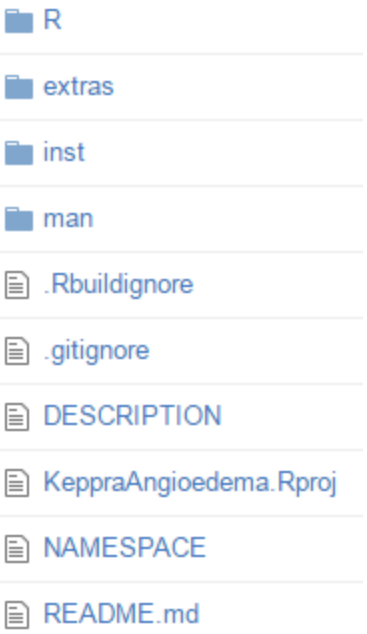
```
sql <- SqlRender::loadRenderTranslateSql("GetDenominator.sql",
                                         "DrugsInPeds",
                                         attr(conn,"dbms"),
                                         oracleTempSchema = oracleTempSchema,
                                         cdm_database_schema = cdmDatabaseSchema,
                                         study_start_date = studyStartDate,
                                         study_end_date = studyEndDate,
                                         split_by_age_group = splitByAgeGroup,
                                         split_by_year = splitByYear,
                                         split_by_gender = splitByGender,
                                         restict_to_persons_with_data = restrictToPersonsWithData,
                                         use_derived_observation_periods = useDerivedObservationPeriods)
denominator <- DatabaseConnector::querySql(conn, sql)
names(denominator) <- SqlRender::snakeCaseToCamelCase(names(denominator))
write.csv(denominator, file = fileName, row.names = FALSE)
```

Study package structure



R contains all user-executable R code

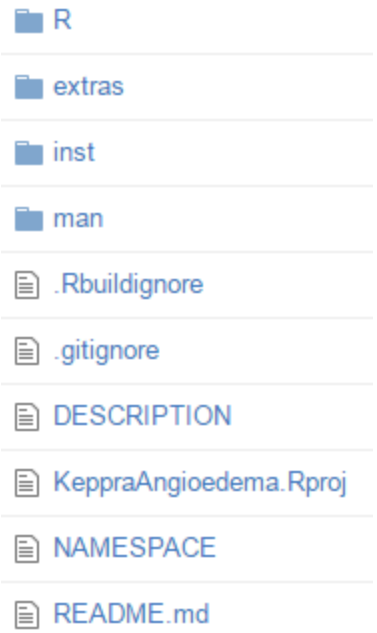
Study package structure



extras contains code used by the study coordinator:

- Code for importing CIRCE definitions
- Testing code
- Code for aggregating data across sites

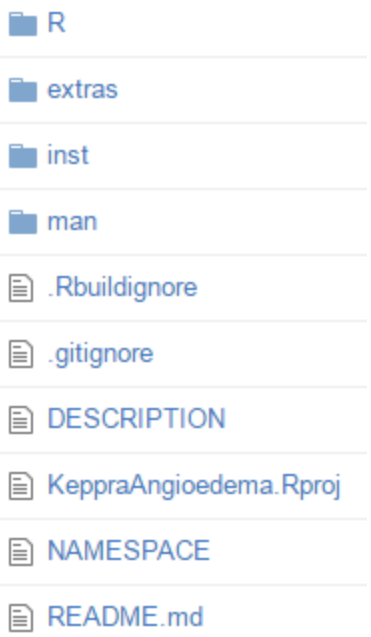
Study package structure



inst is for non-R content of the payload

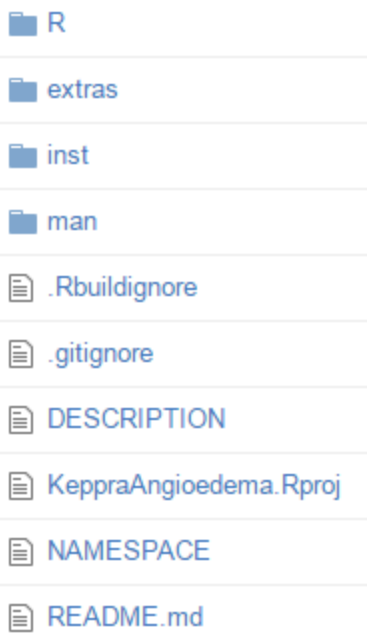
- SQL
- Settings files
- …

Study package structure



DESCRIPTION tells R about package dependencies
(E.g. methods in the methods library)

Study package structure



README.md is what is shown in the GitHub front page

Essentials

Reading

- OHDSI Collobarative Studies FAQ:
  http://www.ohdsi.org/web/wiki/doku.php?id=research:studies:faq
- R packages manual:
  http://r-pkgs.had.co.nz/
- SqlRender vignette:
  https://raw.githubusercontent.com/OHDSI/SqlRender/master/inst/doc/UsingSqlRender.pdf
- OHDSI R and SQL code styles:
  http://www.ohdsi.org/web/wiki/doku.php?id=development:ohdsi_code_style_for_r
  http://www.ohdsi.org/web/wiki/doku.php?id=development:ohdsi_code_style_for_sql

Sites:

- SqlRender test site: http://sqlrenderweb.ohdsi.org:2121/

Contacts:

- schuemie@ohdsi.org
- msuchard@ucla.edu

Conclusions

- Anyone can initiate a study and be a study coordinator
  - Follow the steps described in the FAQ
  - Need to have a clear research question
  - Need someone with R and SQL skills
- OHDSI standards for distributed analysis are evolving
  - Check latest study package for most recent ideas
- Martijn and Marc can help with the writing of the package

Not discussed

- How to find partner sites
  - Public Achilles
- Data quality assurance
  - Sharing Achilles Heel?
- Workflow management
- Governance
  - Data use agreement
  - …