MGRID

Clinical Data Integration

RIMBAA meeting Amsterdam

16 Feb 2011

Henk Enting <h.d.enting@mgrid.net>

Overview

Introduction

Goal

MGRID Approach

Code generation

Conclusions

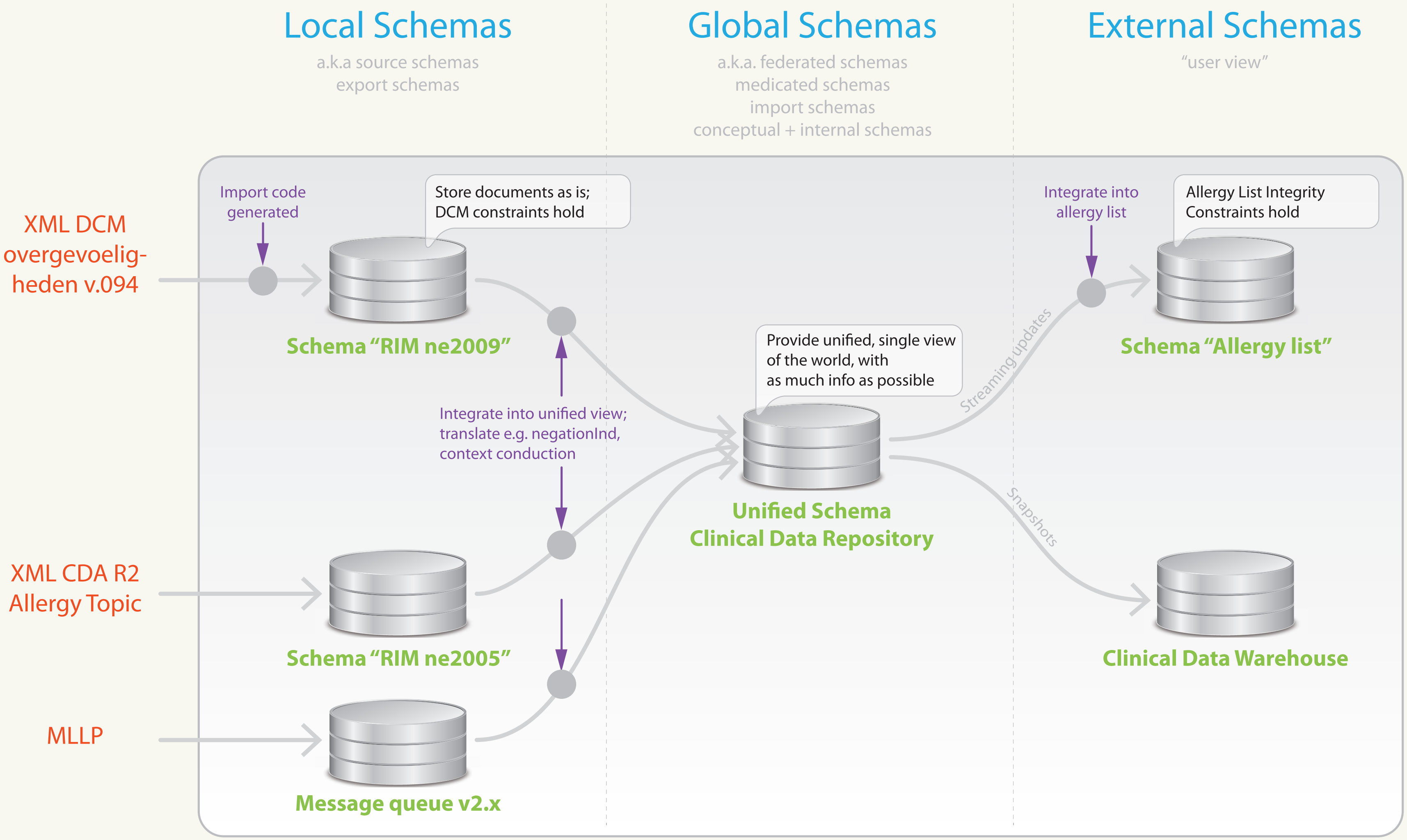
About MGRID

- Spin off of Portavita in 2007
- Goal: create a storage layer specifically designed for eHealth applications
- Funding in 2009 by T-Systems Venture Fund
- 1.0 release nov 2010
- Currently migrating Portavita

MGRID Features

- Distributed relational database
- Powerful constraint and query language with built in:
  - Native support for HL7 R1 and R2 datatypes
  - Codesystem support (e.g. SNOMED CT, HL7v3 codesystems)
- Snapshot support for DWH usage

Clinical Data Integration Framework



Local Schemas

a.k.a source schemas
export schemas

Global Schemas

a.k.a. federated schemas
medicated schemas
import schemas
conceptual + internal schemas

External Schemas

"user view"

XML DCM overgevoeligheden v.094

Import code generated

Store documents as is; DCM constraints hold

Schema "RIM ne2009"

Integrate into unified view; translate e.g. negationInd, context conduction

XML CDA R2 Allergy Topic

Schema "RIM ne2005"

MLLP

Message queue v2.x

Provide unified, single view of the world, with as much info as possible

Unified Schema
Clinical Data Repository

Streaming updates

Snapshots

Integrate into allergy list

Allergy List Integrity Constraints hold

Schema "Allergy list"

Clinical Data Warehouse

The MGRID Clinical Data Platform is the ideal solution for Clinical Data Integration. It supports multiple concurrent schema versions, with powerful primitives such as foreign data wrappers, update triggers and snapshots, as well as database and programming language support for healthcare datatypes.

Updating a Clinical Data Repository

- An update affects multiple RIM tables
- So multiple SQL insert/update statements are necessary, but:
  - The whole message/document should be retrievable (or reproducable)
  - The update should be transactional (performed correctly as a whole or nothing is updated at all)

Challenges

- Different HL7 versions
- Contradicting information
- Overlapping information
- Information with different levels of detail

MGRID Approach

- Multi level: higher levels benefit from powerful primitives on lower level
- Lowest level: relation database augmented with:
  - HL7 R1 and R2 datatypes to avoid cumbersome mappings and explosion of the datamodel
  - Table inheritance to support hierarchical structure of the RIM(s)
  - Database schemas to support different information models concurrently
- E.g. the higher level function dcm_allergy_insert() benefits from a lower level (generated) act_insert() function which benefits from generated RIM DDL, HL7 datatypes

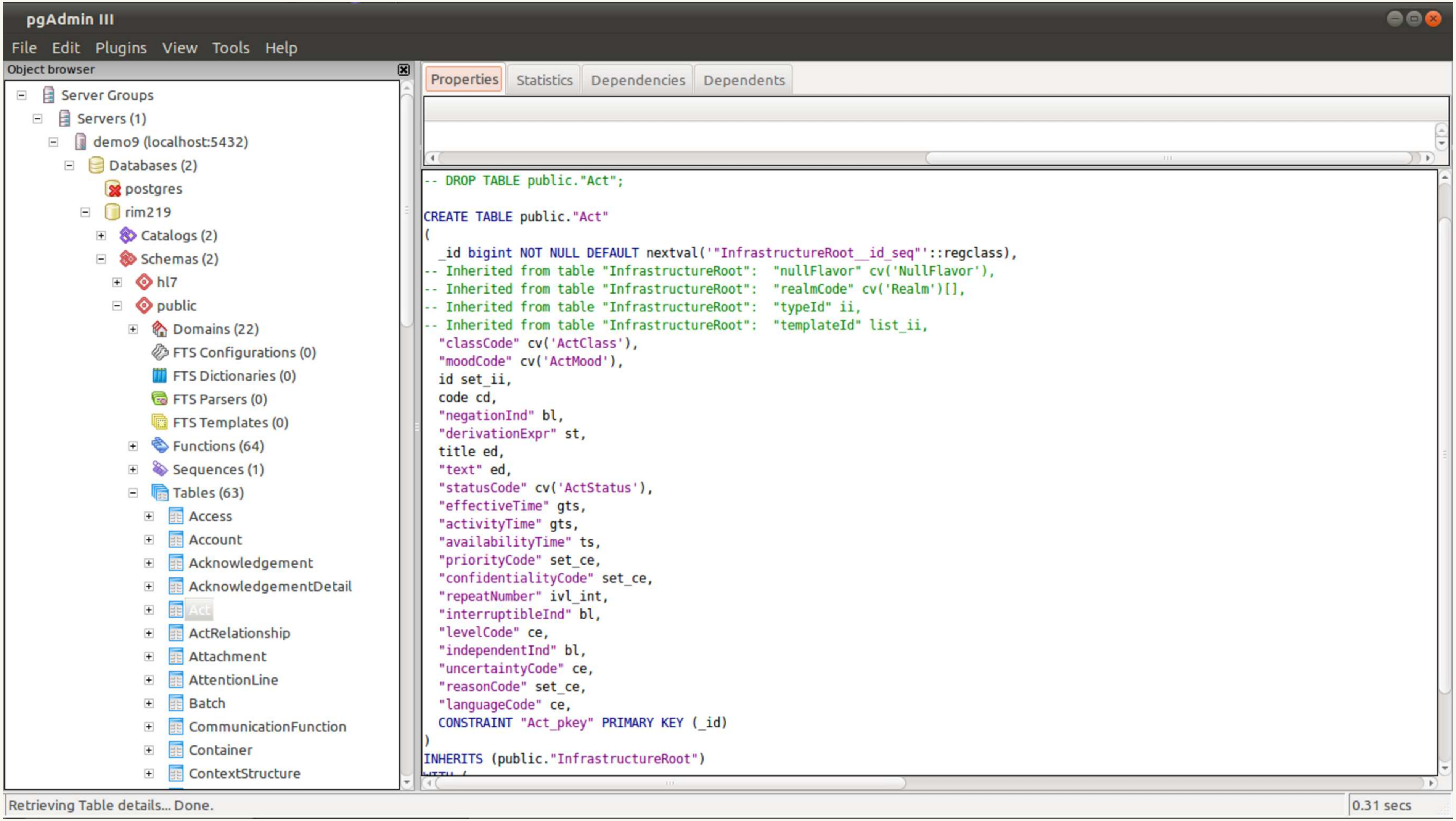
HL7 datatypes

- All R1 and R2 datatypes are available in the database
- This eliminates the need for many datatype related steps from the <HL7 ORM best practices>:
    - Splitting up datatypes into multiple columns: e.g. `effective_time` ⇒ (`effective_time_from`, `effective_time_to`)
    - Expanding datatypes into extra tables: addresses, name parts
- This makes the resulting datamodel a whole lot simpler, and the queries too!

Table inheritance

- MGRID enhanced PostgreSQL table inheritance so it can be used for hierarchical data structures.
- Table per class OR-mapping without the performance hit.
- Eliminates even more steps from the <HL7 ORM best practices>:
  - remove generalization relationships
  - copy inherited attributes to subclasses
  - copy inherited associations to subclasses
- The resulting datamodel is very similar to the hierarchical RIM model.

Datatype and inheritance demo

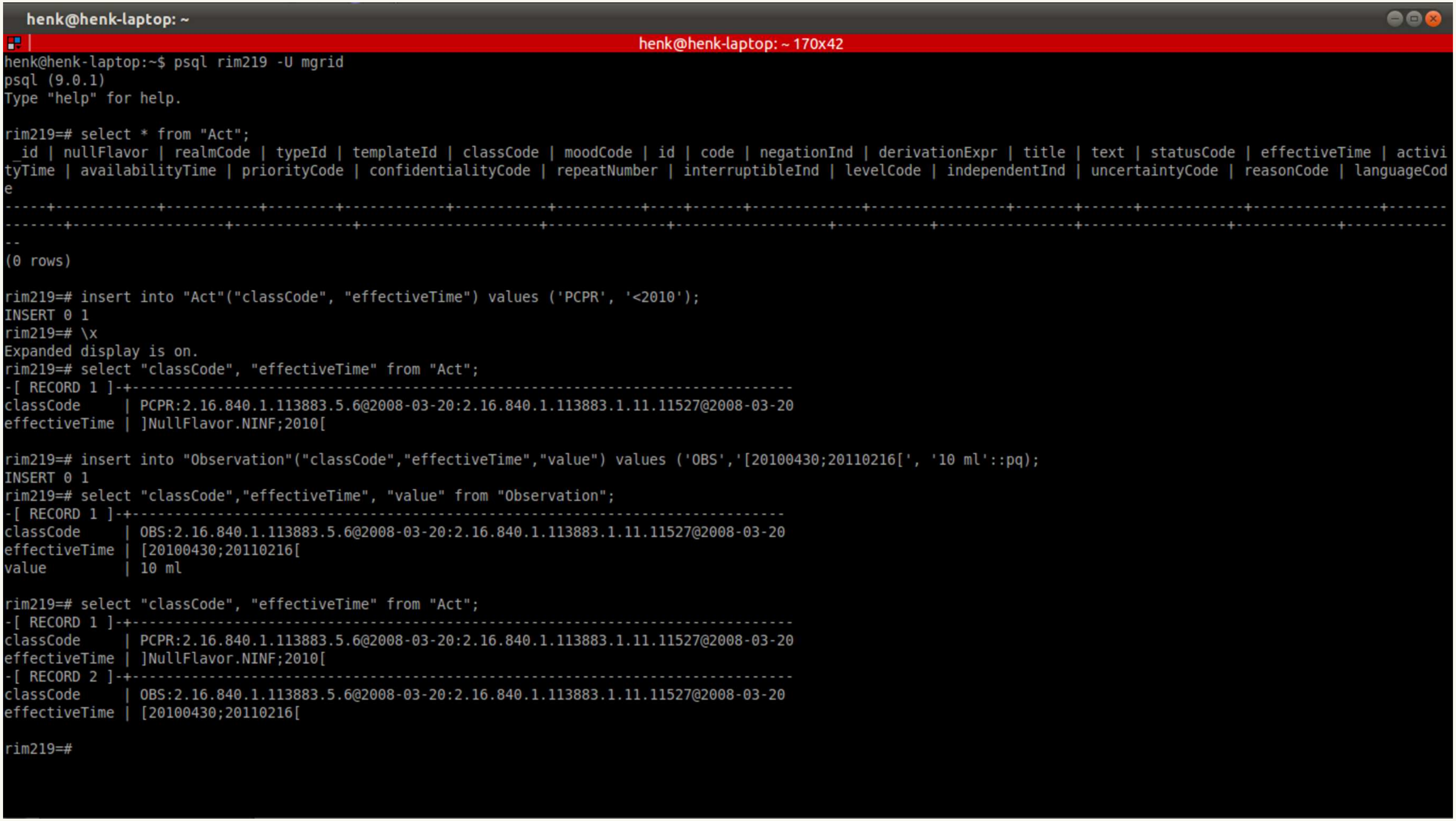


```
-- DROP TABLE public."Act";

CREATE TABLE public."Act"
(
  _id bigint NOT NULL DEFAULT nextval('"InfrastructureRoot__id_seq"'::regclass),
-- Inherited from table "InfrastructureRoot":  "nullFlavor" cv('NullFlavor'),
-- Inherited from table "InfrastructureRoot":  "realmCode" cv('Realm')[],
-- Inherited from table "InfrastructureRoot":  "typeId" ii,
-- Inherited from table "InfrastructureRoot":  "templateId" list_ii,
  "classCode" cv('ActClass'),
  "moodCode" cv('ActMood'),
  id set_ii,
  code cd,
  "negationInd" bl,
  "derivationExpr" st,
  title ed,
  "text" ed,
  "statusCode" cv('ActStatus'),
  "effectiveTime" gts,
  "activityTime" gts,
  "availabilityTime" ts,
  "priorityCode" set_ce,
  "confidentialityCode" set_ce,
  "repeatNumber" ivl_int,
  "interruptibleInd" bl,
  "levelCode" ce,
  "independentInd" bl,
  "uncertaintyCode" ce,
  "reasonCode" set_ce,
  "languageCode" ce,
  CONSTRAINT "Act_pkey" PRIMARY KEY (_id)
)
INHERITS (public."InfrastructureRoot")
```

Retrieving Table details... Done.

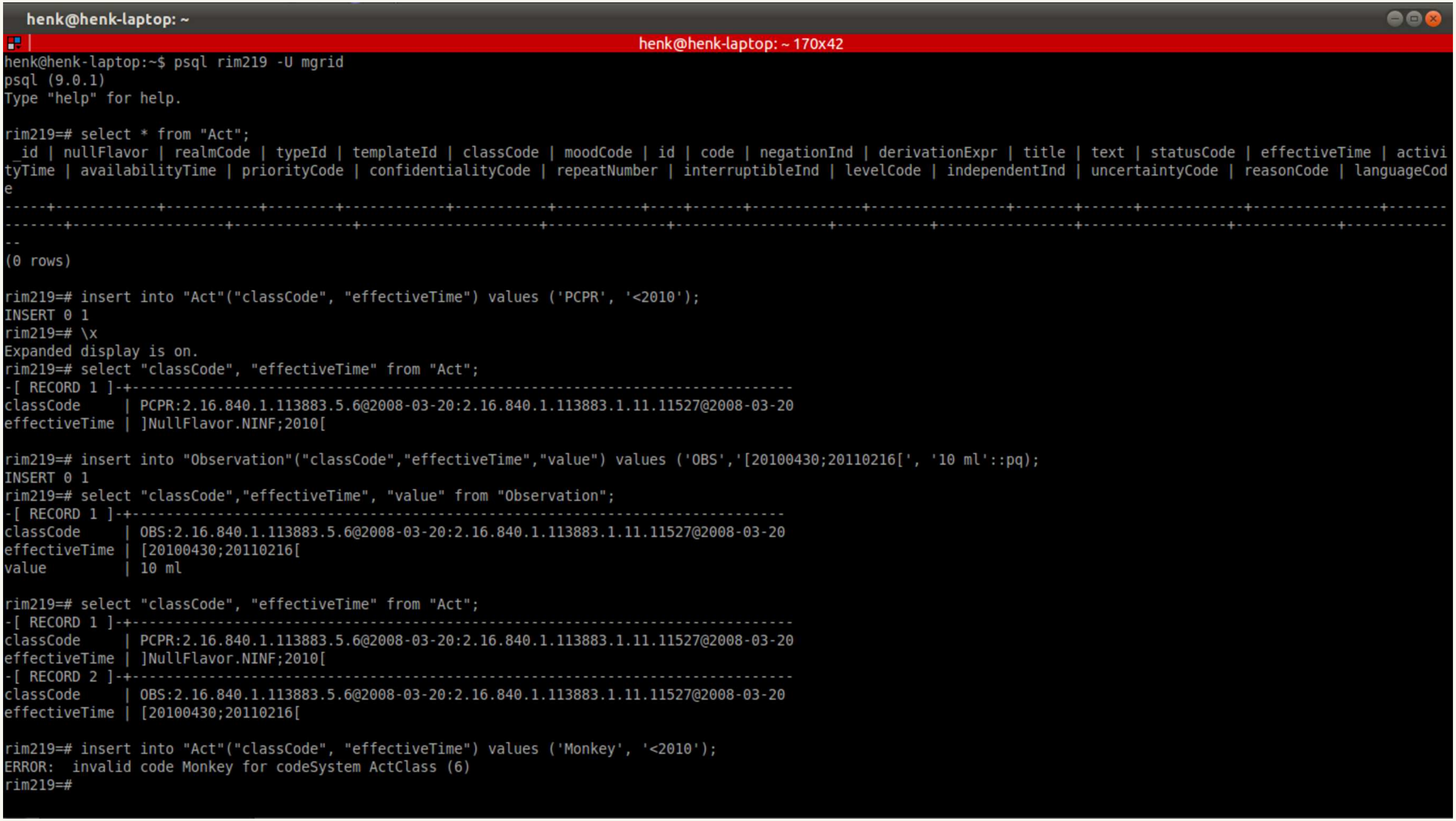
0.31 secs

Datatype and inheritance demo



```
henk@henk-laptop:~$ psql rim219 -U mgrid
psql (9.0.1)
Type "help" for help.

rim219=# select * from "Act";
 _id | nullFlavor | realmCode | typeId | templateId | classCode | moodCode | id | code | negationInd | derivationExpr | title | text | statusCode | effectiveTime | activi
tyTime | availabilityTime | priorityCode | confidentialityCode | repeatNumber | interruptibleInd | levelCode | independentInd | uncertaintyCode | reasonCode | languageCod
e
-----+------------+-----------+--------+------------+-----------+----------+----+------+-------------+----------------+-------+------+------------+---------------+-------
-------+------------------+--------------+---------------------+--------------+------------------+-----------+----------------+-----------------+------------+------------
--
(0 rows)

rim219=# insert into "Act"("classCode", "effectiveTime") values ('PCPR', '<2010');
INSERT 0 1
rim219=# \x
Expanded display is on.
rim219=# select "classCode", "effectiveTime" from "Act";
-[ RECORD 1 ]-+------------------------------------------------------------------------------
classCode     | PCPR:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | ]NullFlavor.NINF;2010[

rim219=# insert into "Observation"("classCode","effectiveTime","value") values ('OBS','[20100430;20110216[', '10 ml'::pq);
INSERT 0 1
rim219=# select "classCode","effectiveTime", "value" from "Observation";
-[ RECORD 1 ]-+-----------------------------------------------------------------------------
classCode     | OBS:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | [20100430;20110216[
value         | 10 ml

rim219=# select "classCode", "effectiveTime" from "Act";
-[ RECORD 1 ]-+------------------------------------------------------------------------------
classCode     | PCPR:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | ]NullFlavor.NINF;2010[
-[ RECORD 2 ]-+------------------------------------------------------------------------------
classCode     | OBS:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | [20100430;20110216[

rim219=#
```

Datatype and inheritance demo

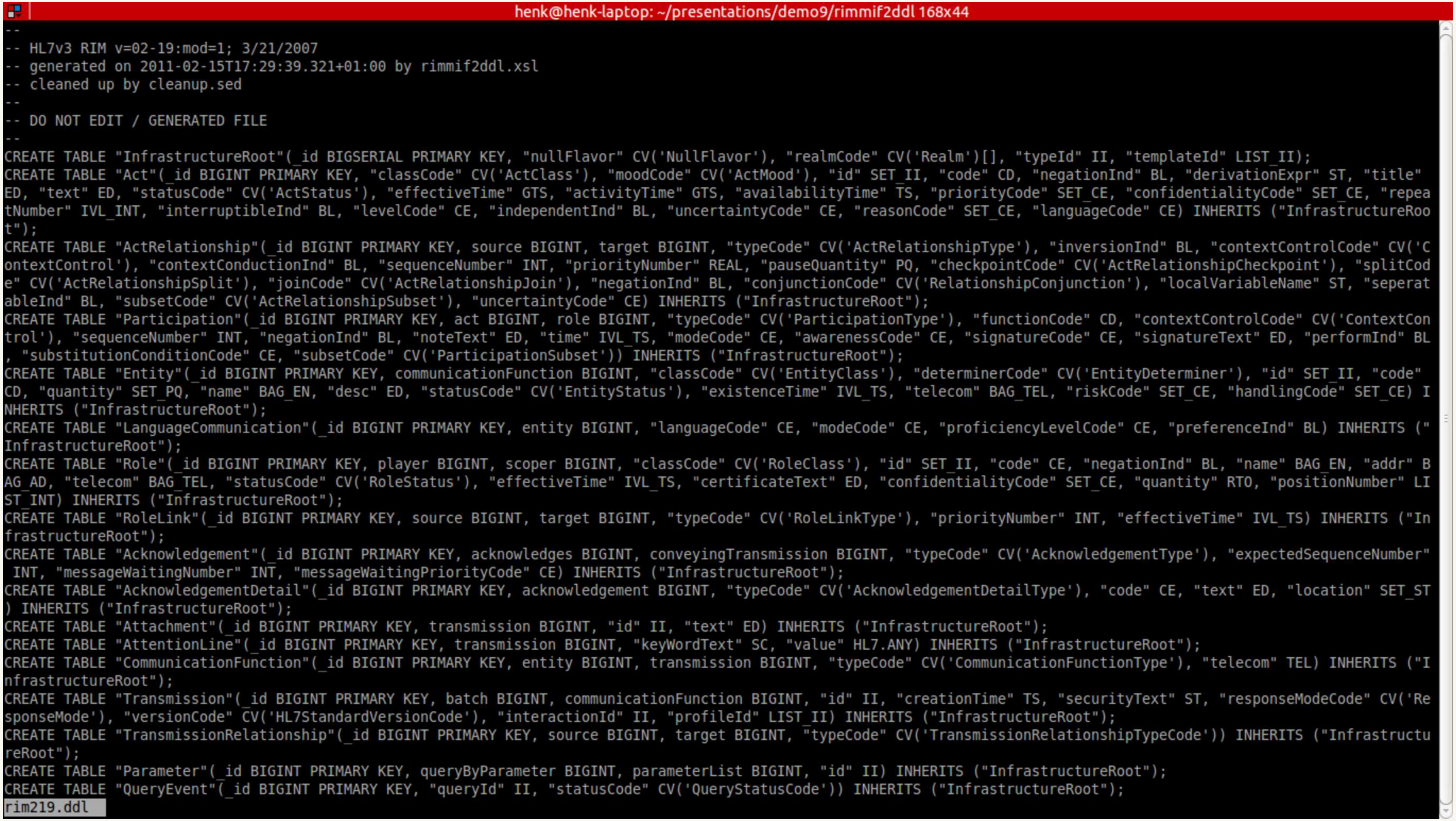


```
henk@henk-laptop:~$ psql rim219 -U mgrid
psql (9.0.1)
Type "help" for help.

rim219=# select * from "Act";
 _id | nullFlavor | realmCode | typeId | templateId | classCode | moodCode | id | code | negationInd | derivationExpr | title | text | statusCode | effectiveTime | activi
tyTime | availabilityTime | priorityCode | confidentialityCode | repeatNumber | interruptibleInd | levelCode | independentInd | uncertaintyCode | reasonCode | languageCod
e
-----+------------+-----------+--------+------------+-----------+----------+----+------+-------------+----------------+-------+------+------------+---------------+-------
-------+------------------+--------------+---------------------+--------------+------------------+-----------+----------------+-----------------+------------+------------
--
(0 rows)

rim219=# insert into "Act"("classCode", "effectiveTime") values ('PCPR', '<2010');
INSERT 0 1
rim219=# \x
Expanded display is on.
rim219=# select "classCode", "effectiveTime" from "Act";
-[ RECORD 1 ]-+----------------------------------------------------------------------------
classCode     | PCPR:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | ]NullFlavor.NINF;2010[

rim219=# insert into "Observation"("classCode","effectiveTime","value") values ('OBS','[20100430;20110216[', '10 ml'::pq);
INSERT 0 1
rim219=# select "classCode","effectiveTime", "value" from "Observation";
-[ RECORD 1 ]-+---------------------------------------------------------------------------
classCode     | OBS:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | [20100430;20110216[
value         | 10 ml

rim219=# select "classCode", "effectiveTime" from "Act";
-[ RECORD 1 ]-+----------------------------------------------------------------------------
classCode     | PCPR:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | ]NullFlavor.NINF;2010[
-[ RECORD 2 ]-+----------------------------------------------------------------------------
classCode     | OBS:2.16.840.1.113883.5.6@2008-03-20:2.16.840.1.113883.1.11.11527@2008-03-20
effectiveTime | [20100430;20110216[

rim219=# insert into "Act"("classCode", "effectiveTime") values ('Monkey', '<2010');
ERROR:  invalid code Monkey for codeSystem ActClass (6)
rim219=#
```

Code Generation

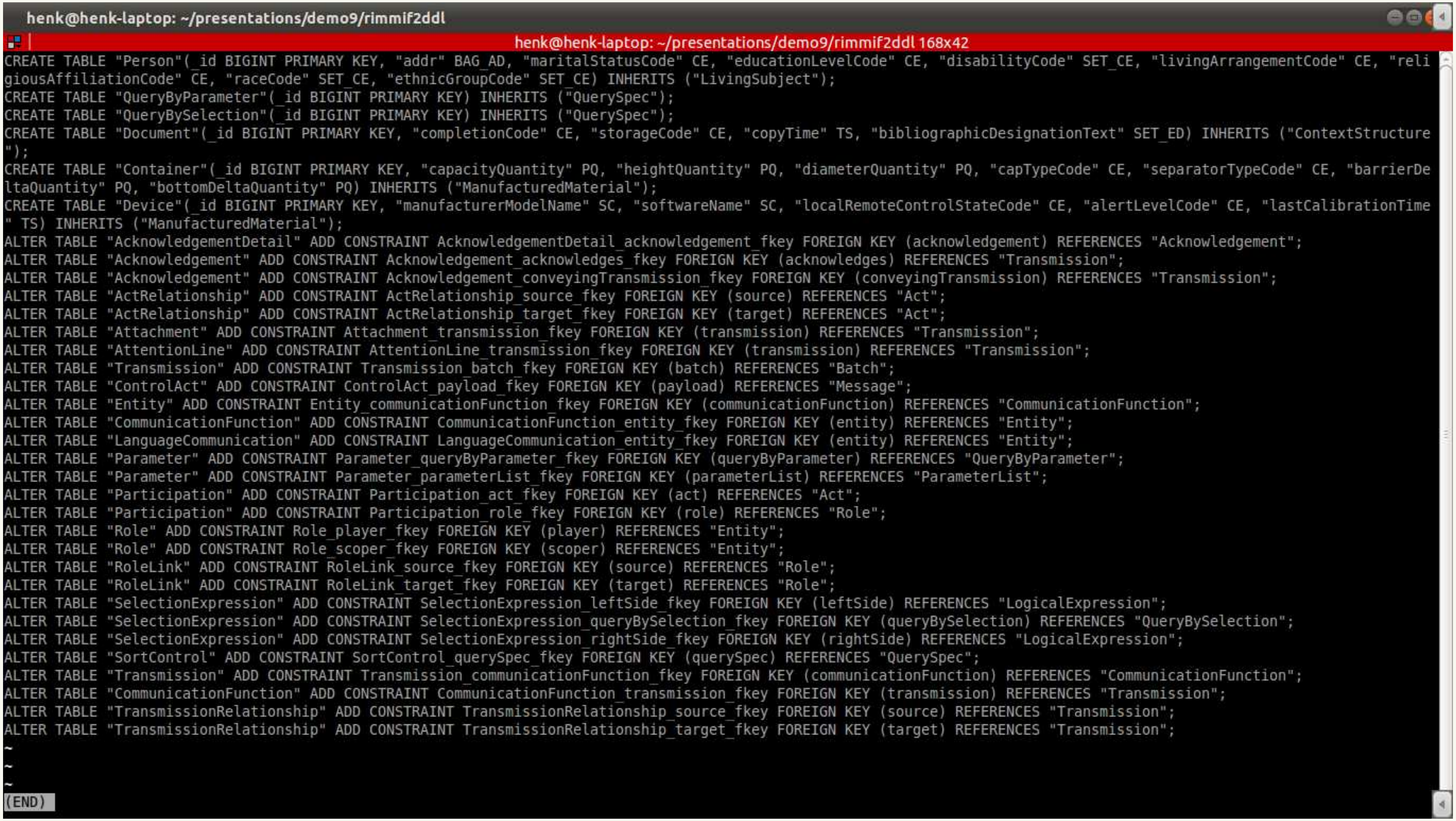
- Generate DDL from RIM MIF
- Generate CRUD operations on RIM classes from table definitions

Generate DDL from RIM: Classes



```
--
-- HL7v3 RIM v=02-19:mod=1; 3/21/2007
-- generated on 2011-02-15T17:29:39.321+01:00 by rimmif2ddl.xsl
-- cleaned up by cleanup.sed
--
-- DO NOT EDIT / GENERATED FILE
--
CREATE TABLE "InfrastructureRoot"(_id BIGSERIAL PRIMARY KEY, "nullFlavor" CV('NullFlavor'), "realmCode" CV('Realm')[], "typeId" II, "templateId" LIST_II);
CREATE TABLE "Act"(_id BIGINT PRIMARY KEY, "classCode" CV('ActClass'), "moodCode" CV('ActMood'), "id" SET_II, "code" CD, "negationInd" BL, "derivationExpr" ST, "title"
ED, "text" ED, "statusCode" CV('ActStatus'), "effectiveTime" GTS, "activityTime" GTS, "availabilityTime" TS, "priorityCode" SET_CE, "confidentialityCode" SET_CE, "repea
tNumber" IVL_INT, "interruptibleInd" BL, "levelCode" CE, "independentInd" BL, "uncertaintyCode" CE, "reasonCode" SET_CE, "languageCode" CE) INHERITS ("InfrastructureRoo
t");
CREATE TABLE "ActRelationship"(_id BIGINT PRIMARY KEY, source BIGINT, target BIGINT, "typeCode" CV('ActRelationshipType'), "inversionInd" BL, "contextControlCode" CV('C
ontextControl'), "contextConductionInd" BL, "sequenceNumber" INT, "priorityNumber" REAL, "pauseQuantity" PQ, "checkpointCode" CV('ActRelationshipCheckpoint'), "splitCod
e" CV('ActRelationshipSplit'), "joinCode" CV('ActRelationshipJoin'), "negationInd" BL, "conjunctionCode" CV('RelationshipConjunction'), "localVariableName" ST, "seperat
ableInd" BL, "subsetCode" CV('ActRelationshipSubset'), "uncertaintyCode" CE) INHERITS ("InfrastructureRoot");
CREATE TABLE "Participation"(_id BIGINT PRIMARY KEY, act BIGINT, role BIGINT, "typeCode" CV('ParticipationType'), "functionCode" CD, "contextControlCode" CV('ContextCon
trol'), "sequenceNumber" INT, "negationInd" BL, "noteText" ED, "time" IVL_TS, "modeCode" CE, "awarenessCode" CE, "signatureCode" CE, "signatureText" ED, "performInd" BL
, "substitutionConditionCode" CE, "subsetCode" CV('ParticipationSubset')) INHERITS ("InfrastructureRoot");
CREATE TABLE "Entity"(_id BIGINT PRIMARY KEY, communicationFunction BIGINT, "classCode" CV('EntityClass'), "determinerCode" CV('EntityDeterminer'), "id" SET_II, "code"
CD, "quantity" SET_PQ, "name" BAG_EN, "desc" ED, "statusCode" CV('EntityStatus'), "existenceTime" IVL_TS, "telecom" BAG_TEL, "riskCode" SET_CE, "handlingCode" SET_CE) I
NHERITS ("InfrastructureRoot");
CREATE TABLE "LanguageCommunication"(_id BIGINT PRIMARY KEY, entity BIGINT, "languageCode" CE, "modeCode" CE, "proficiencyLevelCode" CE, "preferenceInd" BL) INHERITS ("
InfrastructureRoot");
CREATE TABLE "Role"(_id BIGINT PRIMARY KEY, player BIGINT, scoper BIGINT, "classCode" CV('RoleClass'), "id" SET_II, "code" CE, "negationInd" BL, "name" BAG_EN, "addr" B
AG_AD, "telecom" BAG_TEL, "statusCode" CV('RoleStatus'), "effectiveTime" IVL_TS, "certificateText" ED, "confidentialityCode" SET_CE, "quantity" RTO, "positionNumber" LI
ST_INT) INHERITS ("InfrastructureRoot");
CREATE TABLE "RoleLink"(_id BIGINT PRIMARY KEY, source BIGINT, target BIGINT, "typeCode" CV('RoleLinkType'), "priorityNumber" INT, "effectiveTime" IVL_TS) INHERITS ("In
frastructureRoot");
CREATE TABLE "Acknowledgement"(_id BIGINT PRIMARY KEY, acknowledges BIGINT, conveyingTransmission BIGINT, "typeCode" CV('AcknowledgementType'), "expectedSequenceNumber"
 INT, "messageWaitingNumber" INT, "messageWaitingPriorityCode" CE) INHERITS ("InfrastructureRoot");
CREATE TABLE "AcknowledgementDetail"(_id BIGINT PRIMARY KEY, acknowledgement BIGINT, "typeCode" CV('AcknowledgementDetailType'), "code" CE, "text" ED, "location" SET_ST
) INHERITS ("InfrastructureRoot");
CREATE TABLE "Attachment"(_id BIGINT PRIMARY KEY, transmission BIGINT, "id" II, "text" ED) INHERITS ("InfrastructureRoot");
CREATE TABLE "AttentionLine"(_id BIGINT PRIMARY KEY, transmission BIGINT, "keyWordText" SC, "value" HL7.ANY) INHERITS ("InfrastructureRoot");
CREATE TABLE "CommunicationFunction"(_id BIGINT PRIMARY KEY, entity BIGINT, transmission BIGINT, "typeCode" CV('CommunicationFunctionType'), "telecom" TEL) INHERITS ("I
nfrastructureRoot");
CREATE TABLE "Transmission"(_id BIGINT PRIMARY KEY, batch BIGINT, communicationFunction BIGINT, "id" II, "creationTime" TS, "securityText" ST, "responseModeCode" CV('Re
sponseMode'), "versionCode" CV('HL7StandardVersionCode'), "interactionId" II, "profileId" LIST_II) INHERITS ("InfrastructureRoot");
CREATE TABLE "TransmissionRelationship"(_id BIGINT PRIMARY KEY, source BIGINT, target BIGINT, "typeCode" CV('TransmissionRelationshipTypeCode')) INHERITS ("Infrastructu
reRoot");
CREATE TABLE "Parameter"(_id BIGINT PRIMARY KEY, queryByParameter BIGINT, parameterList BIGINT, "id" II) INHERITS ("InfrastructureRoot");
CREATE TABLE "QueryEvent"(_id BIGINT PRIMARY KEY, "queryId" II, "statusCode" CV('QueryStatusCode')) INHERITS ("InfrastructureRoot");
rim219.ddl
```

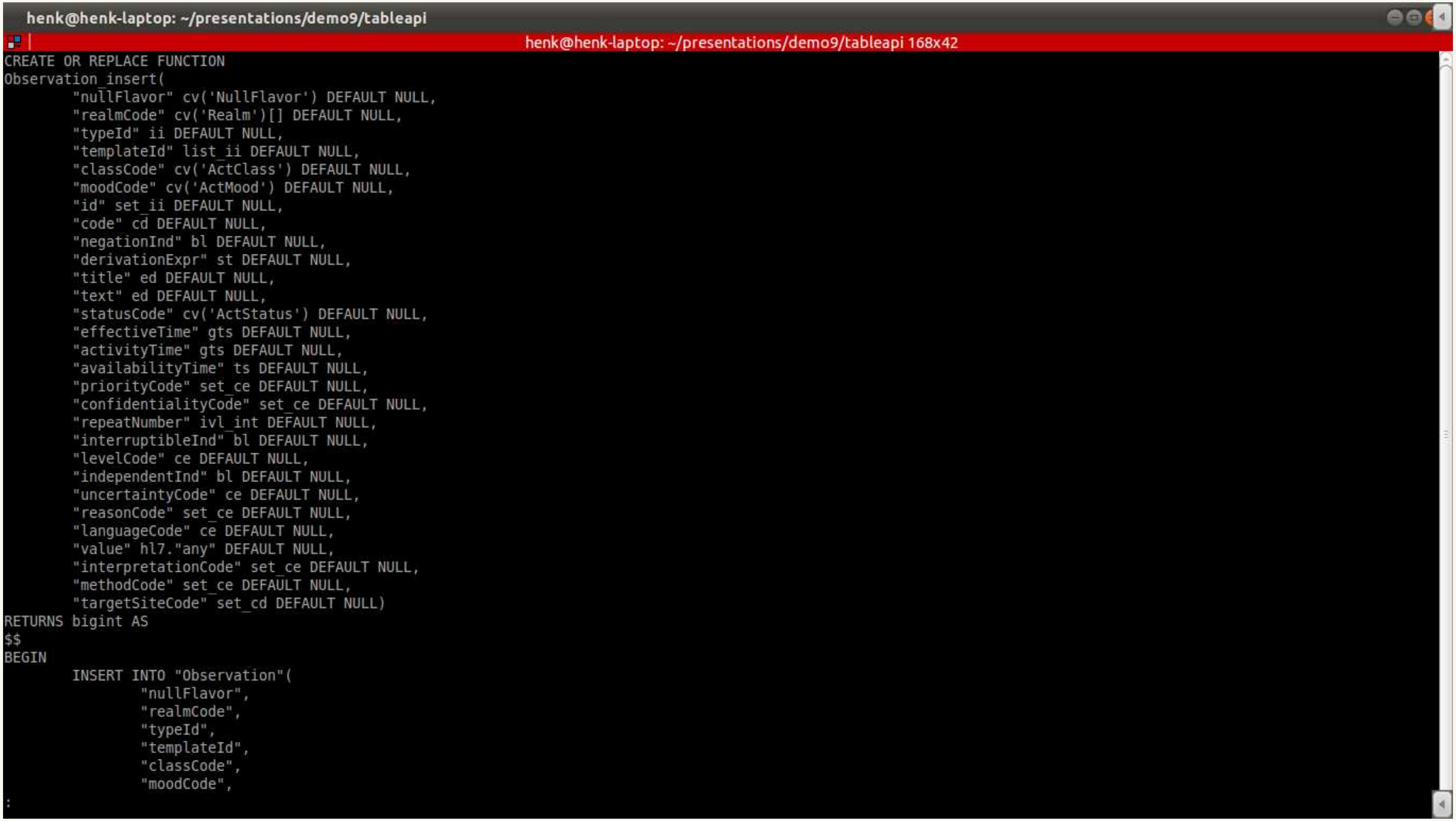
Generate DDL from RIM: Associations



henk@henk-laptop: ~/presentations/demo9/rimmif2ddl
henk@henk-laptop: ~/presentations/demo9/rimmif2ddl 168x42

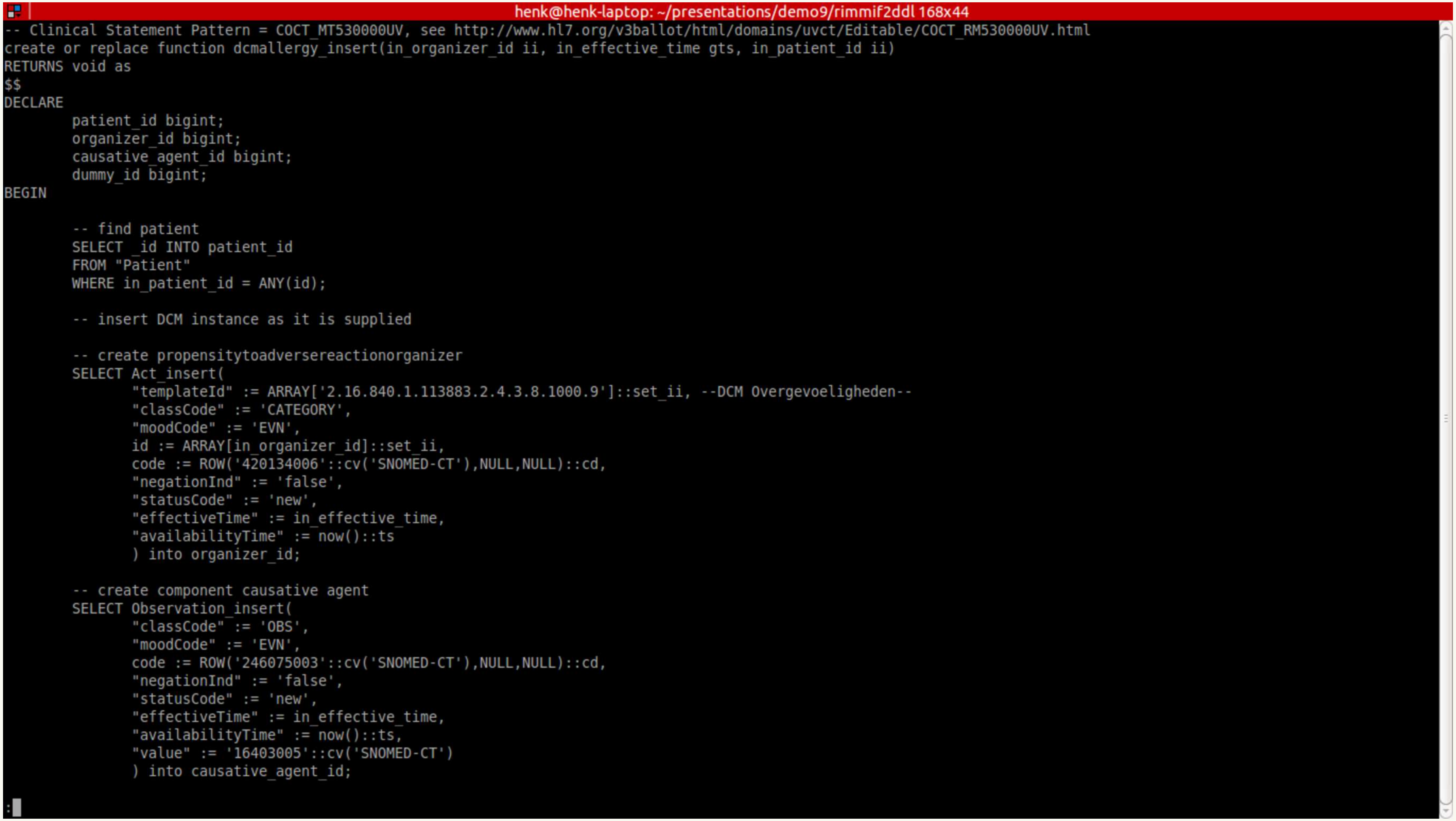
```
CREATE TABLE "Person"(_id BIGINT PRIMARY KEY, "addr" BAG_AD, "maritalStatusCode" CE, "educationLevelCode" CE, "disabilityCode" SET_CE, "livingArrangementCode" CE, "reli
giousAffiliationCode" CE, "raceCode" SET_CE, "ethnicGroupCode" SET_CE) INHERITS ("LivingSubject");
CREATE TABLE "QueryByParameter"(_id BIGINT PRIMARY KEY) INHERITS ("QuerySpec");
CREATE TABLE "QueryBySelection"(_id BIGINT PRIMARY KEY) INHERITS ("QuerySpec");
CREATE TABLE "Document"(_id BIGINT PRIMARY KEY, "completionCode" CE, "storageCode" CE, "copyTime" TS, "bibliographicDesignationText" SET_ED) INHERITS ("ContextStructure
");
CREATE TABLE "Container"(_id BIGINT PRIMARY KEY, "capacityQuantity" PQ, "heightQuantity" PQ, "diameterQuantity" PQ, "capTypeCode" CE, "separatorTypeCode" CE, "barrierDe
ltaQuantity" PQ, "bottomDeltaQuantity" PQ) INHERITS ("ManufacturedMaterial");
CREATE TABLE "Device"(_id BIGINT PRIMARY KEY, "manufacturerModelName" SC, "softwareName" SC, "localRemoteControlStateCode" CE, "alertLevelCode" CE, "lastCalibrationTime
" TS) INHERITS ("ManufacturedMaterial");
ALTER TABLE "AcknowledgementDetail" ADD CONSTRAINT AcknowledgementDetail_acknowledgement_fkey FOREIGN KEY (acknowledgement) REFERENCES "Acknowledgement";
ALTER TABLE "Acknowledgement" ADD CONSTRAINT Acknowledgement_acknowledges_fkey FOREIGN KEY (acknowledges) REFERENCES "Transmission";
ALTER TABLE "Acknowledgement" ADD CONSTRAINT Acknowledgement_conveyingTransmission_fkey FOREIGN KEY (conveyingTransmission) REFERENCES "Transmission";
ALTER TABLE "ActRelationship" ADD CONSTRAINT ActRelationship_source_fkey FOREIGN KEY (source) REFERENCES "Act";
ALTER TABLE "ActRelationship" ADD CONSTRAINT ActRelationship_target_fkey FOREIGN KEY (target) REFERENCES "Act";
ALTER TABLE "Attachment" ADD CONSTRAINT Attachment_transmission_fkey FOREIGN KEY (transmission) REFERENCES "Transmission";
ALTER TABLE "AttentionLine" ADD CONSTRAINT AttentionLine_transmission_fkey FOREIGN KEY (transmission) REFERENCES "Transmission";
ALTER TABLE "Transmission" ADD CONSTRAINT Transmission_batch_fkey FOREIGN KEY (batch) REFERENCES "Batch";
ALTER TABLE "ControlAct" ADD CONSTRAINT ControlAct_payload_fkey FOREIGN KEY (payload) REFERENCES "Message";
ALTER TABLE "Entity" ADD CONSTRAINT Entity_communicationFunction_fkey FOREIGN KEY (communicationFunction) REFERENCES "CommunicationFunction";
ALTER TABLE "CommunicationFunction" ADD CONSTRAINT CommunicationFunction_entity_fkey FOREIGN KEY (entity) REFERENCES "Entity";
ALTER TABLE "LanguageCommunication" ADD CONSTRAINT LanguageCommunication_entity_fkey FOREIGN KEY (entity) REFERENCES "Entity";
ALTER TABLE "Parameter" ADD CONSTRAINT Parameter_queryByParameter_fkey FOREIGN KEY (queryByParameter) REFERENCES "QueryByParameter";
ALTER TABLE "Parameter" ADD CONSTRAINT Parameter_parameterList_fkey FOREIGN KEY (parameterList) REFERENCES "ParameterList";
ALTER TABLE "Participation" ADD CONSTRAINT Participation_act_fkey FOREIGN KEY (act) REFERENCES "Act";
ALTER TABLE "Participation" ADD CONSTRAINT Participation_role_fkey FOREIGN KEY (role) REFERENCES "Role";
ALTER TABLE "Role" ADD CONSTRAINT Role_player_fkey FOREIGN KEY (player) REFERENCES "Entity";
ALTER TABLE "Role" ADD CONSTRAINT Role_scoper_fkey FOREIGN KEY (scoper) REFERENCES "Entity";
ALTER TABLE "RoleLink" ADD CONSTRAINT RoleLink_source_fkey FOREIGN KEY (source) REFERENCES "Role";
ALTER TABLE "RoleLink" ADD CONSTRAINT RoleLink_target_fkey FOREIGN KEY (target) REFERENCES "Role";
ALTER TABLE "SelectionExpression" ADD CONSTRAINT SelectionExpression_leftSide_fkey FOREIGN KEY (leftSide) REFERENCES "LogicalExpression";
ALTER TABLE "SelectionExpression" ADD CONSTRAINT SelectionExpression_queryBySelection_fkey FOREIGN KEY (queryBySelection) REFERENCES "QueryBySelection";
ALTER TABLE "SelectionExpression" ADD CONSTRAINT SelectionExpression_rightSide_fkey FOREIGN KEY (rightSide) REFERENCES "LogicalExpression";
ALTER TABLE "SortControl" ADD CONSTRAINT SortControl_querySpec_fkey FOREIGN KEY (querySpec) REFERENCES "QuerySpec";
ALTER TABLE "Transmission" ADD CONSTRAINT Transmission_communicationFunction_fkey FOREIGN KEY (communicationFunction) REFERENCES "CommunicationFunction";
ALTER TABLE "CommunicationFunction" ADD CONSTRAINT CommunicationFunction_transmission_fkey FOREIGN KEY (transmission) REFERENCES "Transmission";
ALTER TABLE "TransmissionRelationship" ADD CONSTRAINT TransmissionRelationship_source_fkey FOREIGN KEY (source) REFERENCES "Transmission";
ALTER TABLE "TransmissionRelationship" ADD CONSTRAINT TransmissionRelationship_target_fkey FOREIGN KEY (target) REFERENCES "Transmission";
~
~
~
(END)
```

Generate CRUD from table definitions


henk@henk-laptop: ~/presentations/demo9/tableapi
henk@henk-laptop: ~/presentations/demo9/tableapi 168x42

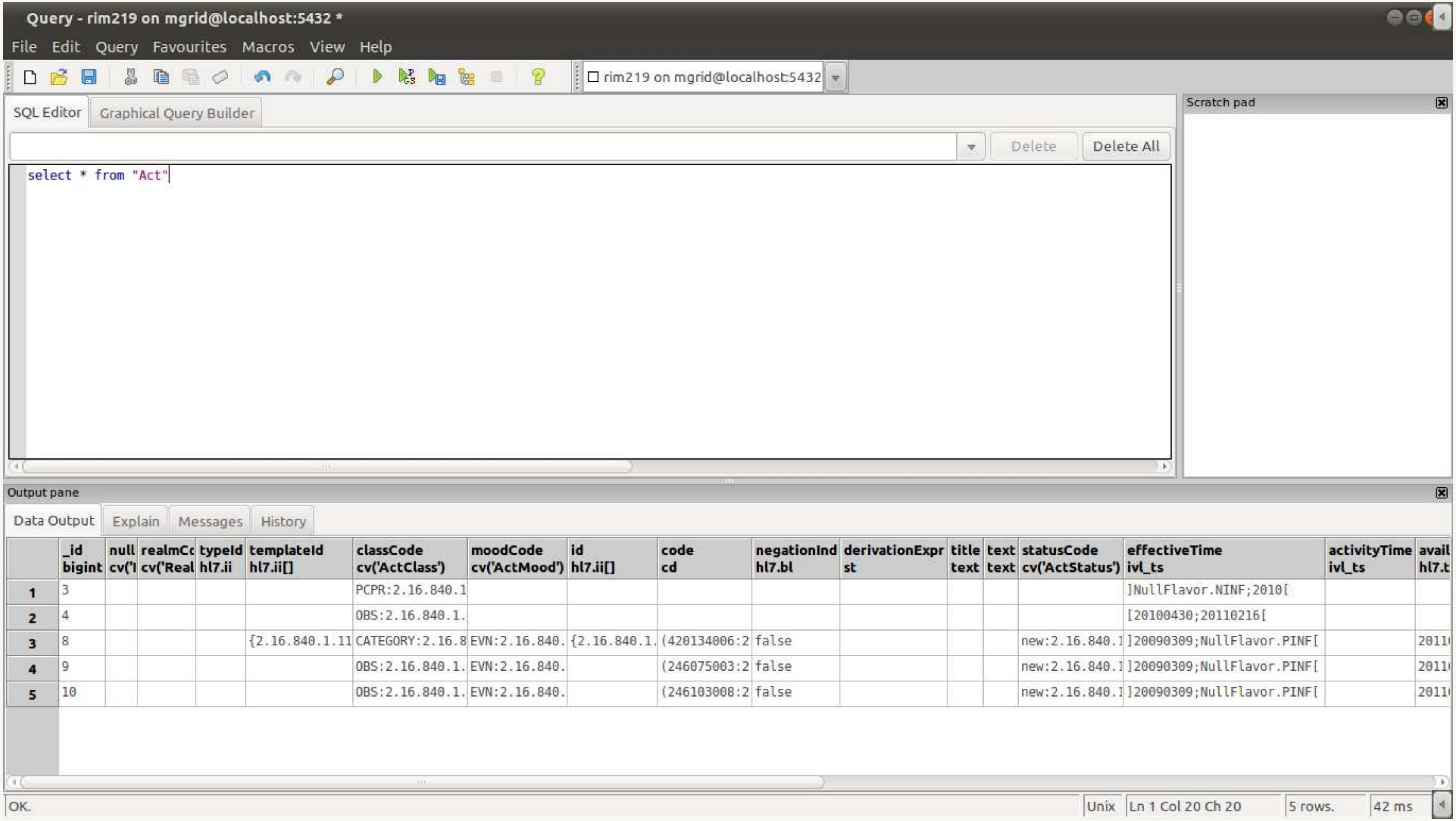
```
CREATE OR REPLACE FUNCTION
Observation_insert(
        "nullFlavor" cv('NullFlavor') DEFAULT NULL,
        "realmCode" cv('Realm')[] DEFAULT NULL,
        "typeId" ii DEFAULT NULL,
        "templateId" list_ii DEFAULT NULL,
        "classCode" cv('ActClass') DEFAULT NULL,
        "moodCode" cv('ActMood') DEFAULT NULL,
        "id" set_ii DEFAULT NULL,
        "code" cd DEFAULT NULL,
        "negationInd" bl DEFAULT NULL,
        "derivationExpr" st DEFAULT NULL,
        "title" ed DEFAULT NULL,
        "text" ed DEFAULT NULL,
        "statusCode" cv('ActStatus') DEFAULT NULL,
        "effectiveTime" gts DEFAULT NULL,
        "activityTime" gts DEFAULT NULL,
        "availabilityTime" ts DEFAULT NULL,
        "priorityCode" set_ce DEFAULT NULL,
        "confidentialityCode" set_ce DEFAULT NULL,
        "repeatNumber" ivl_int DEFAULT NULL,
        "interruptibleInd" bl DEFAULT NULL,
        "levelCode" ce DEFAULT NULL,
        "independentInd" bl DEFAULT NULL,
        "uncertaintyCode" ce DEFAULT NULL,
        "reasonCode" set_ce DEFAULT NULL,
        "languageCode" ce DEFAULT NULL,
        "value" hl7."any" DEFAULT NULL,
        "interpretationCode" set_ce DEFAULT NULL,
        "methodCode" set_ce DEFAULT NULL,
        "targetSiteCode" set_cd DEFAULT NULL)
RETURNS bigint AS
$$
BEGIN
        INSERT INTO "Observation"(
                "nullFlavor",
                "realmCode",
                "typeId",
                "templateId",
                "classCode",
                "moodCode",
:
```

Allergy DCM



```
-- Clinical Statement Pattern = COCT_MT530000UV, see http://www.hl7.org/v3ballot/html/domains/uvct/Editable/COCT_RM530000UV.html
create or replace function dcmallergy_insert(in_organizer_id ii, in_effective_time gts, in_patient_id ii)
RETURNS void as
$$
DECLARE
        patient_id bigint;
        organizer_id bigint;
        causative_agent_id bigint;
        dummy_id bigint;
BEGIN

        -- find patient
        SELECT _id INTO patient_id
        FROM "Patient"
        WHERE in_patient_id = ANY(id);

        -- insert DCM instance as it is supplied

        -- create propensitytoadversereactionorganizer
        SELECT Act_insert(
               "templateId" := ARRAY['2.16.840.1.113883.2.4.3.8.1000.9']::set_ii, --DCM Overgevoeligheden--
               "classCode" := 'CATEGORY',
               "moodCode" := 'EVN',
               id := ARRAY[in_organizer_id]::set_ii,
               code := ROW('420134006'::cv('SNOMED-CT'),NULL,NULL)::cd,
               "negationInd" := 'false',
               "statusCode" := 'new',
               "effectiveTime" := in_effective_time,
               "availabilityTime" := now()::ts
               ) into organizer_id;

        -- create component causative agent
        SELECT Observation_insert(
               "classCode" := 'OBS',
               "moodCode" := 'EVN',
               code := ROW('246075003'::cv('SNOMED-CT'),NULL,NULL)::cd,
               "negationInd" := 'false',
               "statusCode" := 'new',
               "effectiveTime" := in_effective_time,
               "availabilityTime" := now()::ts,
               "value" := '16403005'::cv('SNOMED-CT')
               ) into causative_agent_id;

:
```

Allergy DCM


Query - rim219 on mgrid@localhost:5432 *
File Edit Query Favourites Macros View Help
rim219 on mgrid@localhost:5432
SQL Editor
Graphical Query Builder
Delete
Delete All
Scratch pad
Output pane
Data Output
Explain
Messages
History

```
select * from "Act"
```

	_id	null	realmCc	typeId	templateId	classCode	moodCode	id	code	negationInd	derivationExpr	title	text	statusCode	effectiveTime	activityTime	avail
	bigint	cv('I	cv('Real	hl7.ii	hl7.ii[]	cv('ActClass')	cv('ActMood')	hl7.ii[]	cd	hl7.bl	st	text	text	cv('ActStatus')	ivl_ts	ivl_ts	hl7.t
1	3					PCPR:2.16.840.1									]NullFlavor.NINF;2010[		
2	4					OBS:2.16.840.1.									[20100430;20110216[		
3	8				{2.16.840.1.11	CATEGORY:2.16.8	EVN:2.16.840.	{2.16.840.1.	(420134006:2	false				new:2.16.840.1	]20090309;NullFlavor.PINF[		2011
4	9					OBS:2.16.840.1.	EVN:2.16.840.		(246075003:2	false				new:2.16.840.1	]20090309;NullFlavor.PINF[		2011
5	10					OBS:2.16.840.1.	EVN:2.16.840.		(246103008:2	false				new:2.16.840.1	]20090309;NullFlavor.PINF[		2011

OK. Unix Ln 1 Col 20 Ch 20 5 rows. 42 ms

Conclusions

- HL7 datatypes and table inheritance make HL7 RIM mapping to a relational database very easy.
- Code generation for DDL and CRUD functions is possible in the local schemas.
- Mapping to the unified schema is less trivial.

Questions?