KIM Conversion: Person - KIM

The following is the conversion to migrate the Person table(s)/view(s) to KIM tables. This document does not address AuthZ or AuthN migration.

# Foreign Keys to Person

* The foreign keys can remain unchanged since we will not be removing the Person table in this proposed solution. We may want to look at what references the Person table but does not have a foreign key. I have noticed many potential foreign keys that are “missing” in various tables. Not only would this help integrity-wise but it would also help our database schema make more sense at-a-glance.

# Views into Person

* Rework/Delete Views {OSP$Person, OSP$User}

Reworking these views may be difficult due to the complex KIM structure. For example: Some entity data will need to be stored in a generic KIM table. To get a ssn number for entity ‘1’ (Select EXT\_ID FROM ENTITY\_T entity, KRIM\_ENTITY\_EXT\_ID\_T ext, KRIM\_ENTITY\_EXT\_TYP\_T type WHERE ext .ENTITY\_ID = entity.ENTITY\_ID and ext.EXT\_ID\_TYP\_CD = type.ENT\_TYP\_CD and entity.ENTITY\_ID = type.ENTITY\_ID AND entity.ENTITY\_ID = ‘1’)

Also, in places old KIM and new KIM store data in different formats. For example a compatible view would need to convert state code “MI” to the abbreviation “Michigan” or country code US to USA

# Standard KIM Entity Data

* KIM supports a standard set of information about each entity. Each set of related information is normalized into a separate table(s). Many of these tables also reference type tables not specific to ENTITY. For example: phone information is located in table KRIM\_ENTITY\_PHONE\_T. This table refers to KRIM\_PHONE\_TYP\_T since there can be many types of phone numbers (work, home, etc.).
* The “TYPE” tables are similar to KCs code tables where they come with a standard set of bootstrap data and are easily maintainable. For much of our data migration we will have to decide whether we add a new “TYPE” or use a generic “Other” type which is usually available out-of-the-box.

# EXT KIM Entity Fields

* Not all information currently in the Person table has a corresponding table in KIM. For these cases, KIM provides a table named KRIM\_ENTITY\_EXT\_ID\_T to hold arbitrary String data[[1]](#endnote-1). The following highlights adding the ssn field for entity id ‘1’ to a KIM entity via the EXT tables.

First, add an entry to the KRIM\_EXT\_TYP\_T { EXT\_ID\_TYP\_CD = ‘SSN’, NM= ‘Social Security Number’}. You can also specify an encryption and sort order for fields if needed. This is a global table for KIM not specific to Entity Ext data types.

Second, add an entry to KRIM\_ENTITY\_EXT\_ID\_T {ENTITY\_EXT\_ID\_ID=next sequence, ENTITY\_ID=’1’, ENITY\_ID\_TYP\_CD=’SSN’, EXT\_ID=”123-45-6789”}

# ENTITY PRIVACY IN KIM

* For each Standard KIM Entity Data type KIM allows data suppression via a suppress flag in KRIM\_ENTITY\_PRIV\_PREF\_T. EXT KIM entity data cannot be suppressed in this fashion (although it can be encrypted via an encrypt flag). **Is this a problem for KC? More research will have to be done into the purpose of this table and the suppress flags.**

# GENERAL OPEN POINTS/NOTES/STRANGENESS

* Currently a person can exist in the Person table w/o existing in the KIM\_PERSON table[[2]](#endnote-2). This is similar to an ENTITY existing in KIM without a PRINCIPAL[[3]](#endnote-3).
* Why does KIM have an ENTITY\_TYP\_CD field in many of the KRIM\_ENTITY\_XXX\_T tables? There is already a Foreign Key to the Entity table which also has an ENTITY\_TYP\_CD. This seems redundant/error prone unless there is some other meaning… If this is redundant data then all new entries in these tables should match the KRIM\_ENTITY\_T table’s ENTITY\_TYP\_CD
* KRIM\_ENTITY\_XXX\_T tables have active & default indicators. We should probably have all entries in these tables be active upon data migration. As for default most entries should be default unless multiple entries exist which then we will have to choose.
* The KRIM\_ENTITY\_CACHE\_T seems to be a common set of fields about an entity. Why is this not a view rather than a separate table? All data must be synced with this table or else it will be inconsistent. Part of our migration may be populating this table.
* KIM has a parallel table structure. For Example: KRIM\_ENTITY\_ADDR\_T and KRIM\_PND\_ADDR\_T share the same structure. Geoff believes that these “PND” (pending) tables are for maintenance and we should not need to interact with them. This needs to be confirmed since right now this is an assumption. This seems a little strange without any real advantage over one set of tables with a “pending” flag.

## Field Conversion

|  |  |  |
| --- | --- | --- |
| PERSON | KIM | Notes |
| PERSON\_ID (VARCHAR(10)) | KRIM\_ENTITY\_T.ENTITY\_ID (VARCHAR(40)) |  |
| VER\_NBR | KRIM\_ENTITY\_T.VER\_NBR |  |
| OBJ\_ID | KRIM\_ENTITY\_T.OBJ\_ID |  |
| ACTIVE\_FLG (CHAR(1)) | KRIM\_ENTITY\_T.ACTV\_IND (VARCHAR(1)) |  |
| USER\_NAME | KRIM\_PRNCPL\_T.PRNCPL\_ID | Seems related to KRIM\_PRNCPL\_T.PRNCPL\_NM |
| UPDATE\_TIMESTAMP | N/A |  |
| UPDATE\_USER | KRIM\_ENTITY\_T.LAST\_UPDT\_DT |  |
| ERA\_COMMONS\_USER\_NAME | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| SSN | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| AGE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID | deprecate in favor of KRIM\_ENTITY\_BIO\_T.BIRTH\_DT |
| AGE\_BY\_FISCAL\_YEAR | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID | deprecate in favor of KRIM\_ENTITY\_BIO\_T.BIRTH\_DT |
| EDUCATIONAL\_LEVEL | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| DEGREE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| MAJOR | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| IS\_HANDICAPPED | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| HANDICAP\_TYPE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| IS\_VETERAN | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| VETERAN\_TYPE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| VISA\_CODE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| VISA\_TYPE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| VISA\_RENEWAL\_DATE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| HAS\_VISA | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| SCHOOL | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| YEAR\_GRADUATED | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| DIRECTORY\_DEPARTMENT | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| PRIMARY\_TITLE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| DICTORY\_TITLE | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| VACATION\_ACCURAL | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| IS\_ON\_SABBATICAL | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| ID\_PROVIDED | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| ID\_VERIFIED | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID |  |
| OFFICE\_LOCATION | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID | Assuming this is a name of a location. Do we need this? If so maybe KIM should add a location label field to KRIM\_ADDR\_T |
| SECONDARY\_OFFICE\_LOCATION | KRIM\_ENTITY\_EXT\_ID\_T. EXT\_ID | Assuming this is a name of a location. Do we need this? If so maybe KIM should add a location label field to KRIM\_ADDR\_T |
| FAX\_NUMBER | KRIM\_ENTITY\_PHONE\_T. PHONE\_NBR | Must create a new KRIM\_PHONE\_TYPE\_T column for new phone type or use KRIM\_PHONE\_TYPE\_T . PHONE\_TYPE\_CD ‘OTH’ |
| PAGER\_NUMBER | KRIM\_ENTITY\_PHONE\_T. PHONE\_NBR | Must create a new KRIM\_PHONE\_TYPE\_T column for new phone type or use KRIM\_PHONE\_TYPE\_T . PHONE\_TYPE\_CD ‘OTH’ |
| MOBILE\_PHONE\_NUMBER | KRIM\_ENTITY\_PHONE\_T. PHONE\_NBR | KRIM\_PHONE\_TYPE\_T . PHONE\_TYPE\_CD ‘MBL’ |
| OFFICE\_PHONE | KRIM\_ENTITY\_PHONE\_T. PHONE\_NBR | KRIM\_PHONE\_TYPE\_T . PHONE\_TYPE\_CD ‘WRK’ |
| SECONDARY\_OFFICE\_PHONE | KRIM\_ENTITY\_PHONE\_T. PHONE\_NBR | Must create a new KRIM\_PHONE\_TYPE\_T column for new phone type or use KRIM\_PHONE\_TYPE\_T . PHONE\_TYPE\_CD ‘WRK’ |
| LAST\_NAME | KRIM\_ENTITY\_NM\_T.LAST\_NM | NM\_TYP\_CD = ‘PFRD’ |
| FIRST\_NAME | KRIM\_ENTITY\_NM\_T.FIRST\_NM | NM\_TYP\_CD = ‘PFRD’ |
| MIDDLE\_NAME | KRIM\_ENTITY\_NM\_T.MIDDLE\_NM | NM\_TYP\_CD = ‘PFRD’ |
| SALUTATION | KRIM\_ENTITY\_NM\_T.TITLE\_NM | NM\_TYP\_CD = ‘PFRD’ |
| FULL\_NAME | N/A | Full name is calculated in KIM from the first, middle, last name (not prefix, suffix, etc.) – we could use an ext field or use KIM auto calculation which could be inconsistent with Person data |
| PRIOR\_NAME | KRIM\_ENTITY\_NM\_T.LAST\_NM | Must create a new KRIM\_NM\_TYPE\_T column for new name type or use NM\_TYP\_CD = ‘OTH’. Also must fill in entries for first middle etc. since prior name will be a new row in KRIM\_ENTITY\_NM\_T |
| EMAIL\_ADDRESS | KRIM\_ENTITY\_EMAIL\_T.EMAIL\_ADDR | EMAIL\_TYP\_CD = ‘WRK’ |
| DATE\_OF\_BIRTH | KRIM\_ENTITY\_BIO\_T.BIRTH\_DT |  |
| GENDER (VARCHAR (30)) | KRIM\_ENTITY\_BIO\_T.GNDR\_CD (VARCHAR(1)) | Must convert “Male” & “Female” to “M” & “F” |
| RACE | KRIM\_ENTITY\_BIO\_T.ETHNCTY\_CD | Should convert fine but not sure if the same classifications will be used in new KIM. No code table with various Ethnic groups to confirm. |
| IS\_FACULTY | KRIM\_ENTITY\_AFLTN\_T | AFLTN\_TYP\_CD = “FCLTY” |
| IS\_GRADUATE\_STUDENT\_STAFF | KRIM\_ENTITY\_AFLTN\_T | Must create a new KRIM \_AFLTN\_TYP\_T column for new type or use AFLTN\_TYP\_CD = “STDNT” |
| IS\_RESEARCH\_STAFF | KRIM\_ENTITY\_AFLTN\_T | Must create a new KRIM \_AFLTN\_TYP\_T column for new type |
| IS\_SERVICE\_STAFF | KRIM\_ENTITY\_AFLTN\_T | Must create a new KRIM \_AFLTN\_TYP\_T column for new type |
| IS\_SUPPORT\_STAFF | KRIM\_ENTITY\_AFLTN\_T | AFLTN\_TYP\_CD = “FCLTY” |
| IS\_OTHER\_ACADEMIC\_GROUP | KRIM\_ENTITY\_AFLTN\_T | Must create a new KRIM \_AFLTN\_TYP\_T column for new type |
| IS\_MEDICAL\_STAFF | KRIM\_ENTITY\_AFLTN\_T | Must create a new KRIM \_AFLTN\_TYP\_T column for new type |
| COUNTRY\_OF\_CITIZENSHIP (VARCHAR(30)) | KRIM\_ENTITY\_CTZNSHP\_T .POSTAL\_COUNTRY\_CODE (VARCHAR(2)) | Must convert from description to 2 digit code |
| HOME\_UNIT | N/A | Seems like this field is located in a different set of KIM tables not related to Entity |
| ADDRESS\_LINE\_1 (VARCHAR(80)) | KRIM\_ENTITY\_ADDRESS\_T.ADDR\_LINE\_1 (VARCHAR(50)) | Not long enough to store old KC data. Should KIM make columns longer? |
| ADDRESS\_LINE\_2 (VARCHAR(80)) | KRIM\_ENTITY\_ADDRESS\_T.ADDR\_LINE\_2 (VARCHAR(50)) | Should KIM make columns longer? |
| ADDRESS\_LINE\_3 (VARCHAR(80)) | KRIM\_ENTITY\_ADDRESS\_T.ADDR\_LINE\_3 (VARCHAR(50)) | Should KIM make columns longer? |
| CITY | KRIM\_ENTITY\_ADDRESS\_T.CITY |  |
| COUNTY | N/A | Should KIM add this? |
| STATE (VARCHAR(30)) | KRIM\_ENTITY\_ADDRESS\_T.POSTAL\_STATE\_CD (VARCHAR(2)) | Must convert from description to postal code |
| POSTAL\_CODE | KRIM\_ENTITY\_ADDRESS\_T.POSTAL\_CD |  |
| COUNTRY\_CODE (CHAR(3)) | KRIM\_ENTITY\_ADDRESS\_T.POSTAL\_CNTRY\_CD (VARCHAR(2)) | Must convert from 3 digit code to 2 digit code |
| KIM\_PERSON\_ID | N/A | Entities will referenced to principals in another manner |

# Migration Plan

Now that we have an approach for mapping the old KIM data to new KIM we need to figure out how to use the new KIM in KC. The goal of this plan is to minimize the impact on KC and well as insulate KC from future changes to KIM.

# Person table, BO, repository.xml…

Currently, Person contains a denormalized bunch of data related to Persons. We can keep the Person table but alter it to remove all non-infrastructure related columns. Then add a column to reference the KIM Entity table. This column will not have a foreign key constraint at the database level since KIM can be installed on a separate database than KC. The referential integrity will have to be taken care of at the application level. Below is an example of the new Person table in its entirety:

**CREATE** **TABLE** PERSON ("PERSON\_ID" VARCHAR2(10) **NOT** **NULL** ENABLE,

"ENTITY\_ID" VARCHAR2(40) **NOT** **NULL** ENABLE,

"VER\_NBR" NUMBER(8,0) **DEFAULT** 1 **NOT** **NULL** ENABLE,

"OBJ\_ID" VARCHAR2(36) **DEFAULT** SYS\_GUID() **NOT** **NULL** ENABLE,

**CONSTRAINT** "PK\_PERSON\_KRA" **PRIMARY** **KEY** ("PERSON\_ID") ENABLE);

Then we will keep the Person BO largely unchanged (API wise). We will do this by removing most of the private fields and using delegation. For example:

**public** **class** Person **extends** KraPersistableBOBase **implements** Contactable {

**private** String personId;

**private** String entityId;

**private** org.kuali.rice.kim.bo.entity.KimEntity entity;

**public** String getLastName() {

**return** **this**.entity.getDefaultName().getLastName();

}

}

Don’t forget to cleanup the **toStringMapper()**, **equals(Object o)**, and **hashCode()** methods (if present).

The field mapping is not perfect as shown in the chart above. We have a couple options to correct this situation. We can:

1. Modify the field returned from the entity before returning from the Person. This may be a little more error prone but will limit the impact across the application.
2. We could change client code to deal with different formatted data.

Finally, the repository.xml, Person.xml files will need to get updated to model the new Person BO and Person table.

Important: Since there will not be a foreign key to the KIM entity table, we to handle the situation where the internal entity object in the Person BO is null (not in the database).

This structure has an added benefit that we will NOT have to worry about changed primary keys when doing a data migration. For example: ProposalPerson has a reference to a Person.personId. The ProposalPerson will not need to be updated because the Person.personId is not changing. Internally, the Person will be pointing to an entity but that is not important to ProposalPerson.

It may make sense to consider making Person an interface w/o getters & setters for entity (KIM) related items. We can also create an impl exposing entity getters/setters for JavaBean purposes. This would further encourage KC to stay insulated from KIM artifacts by only depending on the Person interface.

# Entity Maintenance

With the migration, KC will no longer need to handle entity maintenance (for the most part).

**\*Enter reference to loading up maintenance screens\***

With, the proposed design, KC will need a way to “add” a KIM entity as a KC Person. This will be done through a simple maintenance screen which will allow a user to create a Person and associate that Person with an entity. As discussed previously a Person is really just a KC specific wrapper around a KIM entity. The KC PersonMaintenanceDocument.xml should not need to change but the maintainable fields (which should only be the entity id) in the Person.xml will need to be correct.

1. Up to VARCHAR(100) in length [↑](#endnote-ref-1)
2. Old KIM [↑](#endnote-ref-2)
3. New KIM [↑](#endnote-ref-3)