Medición de la Exactitud de Datos en Sistemas Fuentes: Un caso de Estudio‡

Lorena Etcheverry, Salvador Tercia, Adriana Marotta, Verónika Peralta
CSI, Instituto de Computación, Facultad de Ingeniería
Universidad de la República
lorenae@fing.edu.uy, stercia@fing.edu.uy, amarotta@fing.edu.uy, vperalta@fing.edu.uy
Abril 2007

Resumen: El primer paso necesario para conocer la calidad de la información en un sistema de información multi-fuente es analizar la calidad de los datos contenidos en las fuentes de donde se extrae la información. Este trabajo tiene como objetivo medir la calidad de los datos fuente en un sistema de Data Warehousing (DW) particular: el DW de Enseñanza de la Facultad de Ingeniería. Como propiedad de calidad se trabaja con la dimensión de calidad *exactitud*. Se estudian las características de los datos fuentes, determinando para cada caso la forma más adecuada de medir la exactitud. Luego se implementan los procedimientos correspondientes para realizar la medición y por último se ejecuta la misma.

Palabras clave: Calidad de datos, Exactitud, Medición de Calidad, Data Warehouse

‡ Este trabajo fue parcialmente financiado por Comisión Sectorial de Investigación Científica, Universidad de la República, Montevideo, Uruguay

1 Introducción

En sistemas de información que integran datos provenientes de múltiples fuentes, la calidad de los datos ofrecidos al usuario final depende fuertemente de la calidad de los datos en dichas fuentes. En este contexto, el primer paso necesario para conocer la calidad de los datos entregados a los usuarios es analizar la calidad de los datos contenidos en las fuentes y para esto se deben realizar mediciones de calidad en cada una de las fuentes. Estas mediciones pueden ser realizadas a través de distintos procedimientos, que pueden variar considerablemente según los aspectos de la calidad que se deseen medir.

Tradicionalmente la calidad de los datos se define o caracteriza por medio de múltiples dimensiones, como por ejemplo, frescura, exactitud, completitud, disponibilidad, confianza, etc. Muchos trabajos de investigación definen y modelan algunas dimensiones de calidad, por ejemplo [Wan96][Red96][Jar97]. En particular, en [Wan96] se presenta un estudio de las dimensiones desde la perspectiva de los usuarios. Como las necesidades de los usuarios en términos de calidad pueden ser muy diferentes de una aplicación a otra, cada dimensión de calidad puede representar diferentes visiones o aspectos de la calidad para diferentes usuarios. Por tal razón, suelen definirse sub-dimensiones (llamadas factores de calidad), cada una describiendo un aspecto específico sobre la calidad de los datos.

Respecto a la medición de la calidad en Sistemas de Información, algunos trabajos analizan y clasifican tipos de métricas, unidades y técnicas de medición [Pip02][Nau00][Bal98]. Por ejemplo, en [Pip02] se analizan "percepciones subjetivas" y "mediciones objetivas" de la calidad. Las percepciones subjetivas reflejan necesidades y experiencias de las personas que trabajan con los datos, mientras que las evaluaciones objetivas involucran métricas para los datos. Entre otras cosas, los autores presentan métricas a utilizar para distintas dimensiones de calidad. Para algunas dimensiones de calidad, existen propuestas de métricas para aplicaciones específicas, por ejemplo en [Lab05] se analiza la medición de la exactitud de datos relativos a clientes en aplicaciones tipo CRM (gestión de clientes).

Dada la existencia de numerosas propuestas teóricas que analizan los problemas de calidad de los datos, definen dimensiones de calidad, y estudian como medirlas, nos parece necesario y de gran interés realizar una experimentación de estas propuestas en un caso real. Creemos que realizar las mediciones en casos reales puede generar por lo menos tres aportes al estudio de la calidad: (i) la identificación de problemas concretos que pueden no haber surgido en un estudio teórico, (ii) la validación de la aplicabilidad (o no) de las técnicas de medición propuestas teóricamente, y (iii) el surgimiento de nuevas ideas generadas a partir de problemas concretos de calidad. En este trabajo tomamos como caso de estudio el sistema de DW de la Facultad de Ingeniería, el cual maneja información sobre enseñanza, conteniendo datos sobre estudiantes, materias, docentes, actividades de los estudiantes en la facultad, etc. En [Etc03] se encuentra la descripción detallada del diseño de este sistema, en particular los diagramas de flujo de datos desde que los datos son extraídos de las fuentes hasta que se cargan en el DW.

Respecto a las dimensiones de calidad, consideramos la dimensión *exactitud*, analizamos factores y métricas dentro de esta dimensión y realizamos su medición en las fuentes del DW. La elección y definición de los factores y métricas de exactitud se realiza teniendo en cuenta el caso particular elegido, principalmente los tipos de información contenidos en las fuentes y los tipos de "suciedad" encontrados en sus datos.

Para lograr una medición significativa, útil y eficaz, se trabaja en base a tres tareas principales: (i) A partir del conocimiento de las fuentes se decide qué factores de exactitud se miden en cada tabla, a partir de ellos se determinan las restricciones que deben cumplir los datos y los cálculos que se deben realizar para medirlos. (ii) Se define un conjunto de funciones que permiten realizar los cálculos mencionados anteriormente, especificando las medidas a obtener en cada caso. (iii) Se diseña e implementa un mecanismo que permite automatizar parcialmente el cálculo. Esto implica implementar las funciones de cálculo y definir metadatos que permitan registrar todo lo relativo a la medición, como por ejemplo las funciones utilizadas para los cálculos y los resultados obtenidos.

El resto del reporte se organiza de la siguiente forma: En la sección 2 se presentan consideraciones generales al proceso de medición y en la sección 3 se brinda información acerca del marco teórico del trabajo. En la sección 4 se presentan los mecanismos propuestos para realizar la medición, detallando los tipos de errores a medir en cada tabla. La implementación de dichos mecanismos se presenta en la sección 5, comentando los problemas encontrados. En la sección 6 se detallan los resultados obtenidos y por último, en la sección 7, se presentan las conclusiones.

2 Algunas consideraciones previas

En esta sección se comentan algunos conceptos que consideramos es importante aclarar antes de presentar el trabajo de medición de calidad realizado. Específicamente discutimos sobre la granularidad de la medición y sobre los datos para los que se realizará la medición.

2.1 Granularidad de la medición

Llamamos **granularidad** de la medición a la unidad básica de información a la que asociamos las medidas. La granularidad puede ser: fuente, conjunto de tablas, tabla, tupla, atributo o celda. Por ejemplo si estamos manejando granularidad tabla, las medidas que obtenemos están asociadas a la tabla entera (no a elementos contenidos en la tabla) y si manejamos granularidad celda, las medidas que se obtienen corresponden a celdas concretas (un valor de un atributo de una tabla).

Le llamamos **agregación** de las medidas al pasaje de una granularidad a otra granularidad mayor, es decir con menor nivel de detalle. Por ejemplo, realizamos una agregación cuando, a partir de medidas a nivel de las tuplas de una tabla, calculamos medidas globales para dicha tabla.

2.2 Relevancia y pertinencia de la medición

Cuando se plantea realizar mediciones de un factor de calidad para un atributo, pueden darse las siguientes situaciones:

- Resulta interesante medir dicho factor de calidad sobre dicho atributo, para lo cual es necesario diseñar un procedimiento.
- Resulta interesante medir dicho factor de calidad sobre dicho atributo pero no es necesario utilizar un procedimiento de medición, dado que es posible deducir las medidas a partir de los metadatos de las bases fuente. Por ejemplo, sabemos que ciertos atributos, por construcción, siempre van a cumplir ciertas restricciones de formato.
- No resulta interesante medir dicho factor de calidad sobre dicho atributo. Esto puede ser, por ejemplo, porque constituye un caso análogo a otros.
- No tiene sentido medir dicho factor de calidad sobre dicho atributo. Por ejemplo, en la tabla BD_INSTITUTOS, que contiene información sobre los institutos de la facultad, el atributo IN_CODINST (clave primaria) es un identificador autogenerado que no tiene ningún significado para los usuarios, los cuales se refieren a los institutos por su nombre. Por lo tanto en este caso, no tiene sentido medir si los valores de dicho atributo corresponden con la realidad[1].

En la siguiente sección se enumeran los errores encontrados en las bases fuente, indicando la granularidad adecuada para medirlos y, cuando corresponde, comentando si es relevante y pertinente realizar mediciones.

3 Tipos de errores considerados

En esta sección se describen los tipos de errores[2] considerados, clasificándolos de acuerdo a cuatro factores: correctitud semántica, correctitud sintáctica, consistencia y precisión. Las definiciones de estos factores fueron realizadas tomando como base las que se encuentran en [Per06][Ter05], adaptando las mismas a las características e intereses de nuestra realidad.

1. **Correctitud semántica**: Se refiere al grado de correctitud o validez de los datos [Wan96][Pip02], describiendo qué tan bien representan los conceptos del mundo real. Para nosotros este factor indica si el dato medido es el correspondiente al evento u objeto de la realidad al que está asociado en la base de datos. Por ejemplo, es lo que se mide al verificar si la dirección de un estudiante es la correcta en la realidad, o si los estudiantes registrados como asistentes a un curso son los que asisten realmente. Este

[1] Esta medición corresponde al factor de calidad *correctitud semántica* que se define en la próxima sección.
[2] Usamos el término *error* en forma amplia, incluyendo no sólo valores incorrectos o mal formateados sino también valores que no respetan las preferencias de los usuarios (por ejemplo valores que no siguen una cierta representación estándar).

concepto de correctitud implica una comparación con el mundo real o con un referencial considerado como correcto.

2. **Correctitud sintáctica**: Se refiere a que los datos estén libres de errores sintácticos, tales como errores de escritura [Nau99] o discordancias de formato. Los datos se consideran correctos sintácticamente si satisfacen reglas o restricciones de sintaxis, definidas por la persona que mide el factor. Ejemplos de reglas son: "los números de los salones de clase tienen 3 dígitos" o "los nombres de las calles deben estar registrados en un catálogo de calles".
3. **Consistencia**: Indica si los datos satisfacen reglas de integridad [Red96]. Esta consistencia puede involucrar distintos atributos dentro de la misma tupla, distintas tablas, etc. Ejemplos de reglas son "la edad de los estudiantes debe pertenecer a cierto rango", "los códigos de las asignaturas deben ser únicos" o "cada docente debe pertenecer a un único instituto".
4. **Precisión**: Se refiere al nivel de detalle utilizado en la representación de los datos [Bou02]. Por ejemplo, la fecha de nacimiento de una persona puede representarse indicando el año de su nacimiento, (Ej.:1977), el año y mes de nacimiento (Ej.: julio de 1977), la fecha completa (Ej.: 14/7/1977) o incluso con fecha y hora de nacimiento. Este factor lo mediremos con respecto a una precisión considerada como correcta. En este caso, algunos niveles de precisión pueden considerarse mejores que otros según si a partir de ellos es posible obtener el nivel correcto.

A continuación describimos los errores encontrados en los datos, agrupándolos de acuerdo a los factores descritos.

3.1 Errores de correctitud semántica

En esta categoría encontramos valores que no se corresponden con la realidad o que no se corresponden con lo almacenado en referenciales (que contienen los valores considerados correctos). En este sentido, un referencial puede verse como una representación de la realidad.

En algunos casos, tal como se vio en la sección 2, no es pertinente medir la correctitud semántica de cierto atributo, dado que no existe un dato correspondiente en la realidad. Un ejemplo de esta situación son los campos auto numéricos utilizados como clave primaria. En estos casos puede asignarse un valor neutro de correctitud semántica (por ejemplo el valor 1) para cada una de las tuplas a los efectos de combinarlo con otras mediciones en pasos posteriores.

Los errores de correctitud semántica serán medidos a nivel de celda contra un referencial. Para esto se define la función:

CHECK_REF(attribute, set): Esta función evalúa la existencia o no de los valores de un atributo en un conjunto o referencial

3.2 Errores de correctitud sintáctica

En esta categoría encontramos valores que están fuera de un rango o conjunto determinado, valores que no cumplen con las reglas de formato definidas y valores nulos o por defecto.

Los errores de correctitud sintáctica serán medidos a nivel de celda para lo cual se define la siguiente función:

CHECK_RULE(attribute, rule): Esta función evalúa si los valores de un atributo cumplen con la regla definida. Dicha regla puede especificar un formato propiamente dicho o un conjunto de valores permitidos.

Dentro de los errores de correctitud sintáctica encontramos como caso particular el chequeo de valores nulos. Dada la frecuencia con que necesitamos controlar este tipo de errores definimos la siguiente función específica:

CHECK_NULL(attribute): Esta función evalúa si los valores de un atributo cumplen la restricción de no ser nulos.

3.3 Inconsistencias

En esta categoría encontramos valores que violan restricciones de integridad definidas. Estas restricciones pueden involucrar valores de atributos de una misma tupla, de distintas tuplas en una misma tabla o en tablas distintas. Consideramos que es posible medir este factor en alguno de los siguientes niveles:

Celda	Cuando en la medición de la inconsistencia sólo interviene una celda por tupla. EJ. Chequear si el valor de un atributo es nulo.
Atributo	Cuando en la medición de la inconsistencia intervienen todas las celdas de un atributo EJ: Chequear la unicidad de los valores de un atributo
Tupla	Cuando en la medición de la inconsistencia interviene mas de una celda por tupla. Debe existir una relación conocida entre celdas EJ: edad y fecha de nacimiento
Conjunto de tablas	Cuando en la medición de la inconsistencia intervienen celdas de tuplas de diferentes tablas. EJ: Chequear restricciones de clave foránea.

Tabla 1 - Niveles en la medición de inconsistencias

Para medir inconsistencias generales definimos la siguiente función, con el objetivo de soportar la medición en cualquiera de los niveles de granularidad antes especificados:

CHECK_CONSTRAINT(attribute_set1, attribute_set2, constraint): Esta función evalúa si los valores de un conjunto de atributos (*attribute_set1*) cumplen cierta condición (*constraint*) con respecto a los valores de otro conjunto de atributos (*attribute_set2*)

Dentro de las restricciones a considerar encontramos dos casos particulares: restricciones de unicidad y restricciones de clave foránea. Una restricción de unicidad para un atributo de una tabla implica que todas las tuplas de dicha tabla deben tener valores distintos para el atributo en cuestión, mientras que una restricción de clave foránea entre tablas implica que un conjunto de atributos de una de las tablas debe ser clave en la segunda. Dada la frecuencia con que necesitamos controlar este tipo de restricciones definimos las siguientes funciones específicas:

CHECK_UNIQUE(attribute_set): Esta función evalúa la unicidad de los valores de un conjunto de atributos en una tabla.

CHECK_FK(attribute_set1, attribute_set2): Esta función evalúa que los valores de un conjunto de atributos se encuentren dentro de los valores de otro conjunto de atributos de otra tabla.

3.4 Errores de precisión

En esta categoría encontramos valores que no tienen el nivel de precisión requerido, representado por medio de un referencial. Para medir los errores de precisión se define la función *CHECK_LEVEL(attribute, set)*. Esta función determina el nivel de precisión de cada uno de los valores de un atributo, lo compara con el nivel de precisión requerido y le asigna un valor de exactitud resultante de esta comparación.

A continuación presentamos un resumen de los tipos de errores encontrados:

Correctitud semántica	**Correctitud sintáctica**	**Consistencia**	**Precisión**
Valor no se corresponde con la realidad	Valor fuera de rango	No se cumple dependencia entre valores de atributos (de una misma tupla o de tuplas diferentes)	Valor que no tiene suficiente nivel de precisión.

	Valor que no cumple con la regla de formato (gramática) del atributo	No se cumple una clave foránea	
	Valores nulos o por defecto	No se cumple restricción de unicidad	

Tabla 2– Tipos de errores encontrados

4 Diseño de la medición

A continuación presentamos una descripción de cada tabla de la base fuente, detallando luego que factores se midieron en cada una de ellas, qué funciones de las definidas en la sección 3 se aplicaron para realizar la medición de los mismos y con que granularidad se realizó la medición.

4.1 Descripción de las tablas del sistema fuente

En esta sección se listan las tablas consideradas del sistema fuente y sus atributos. Una descripción más detallada de las mismas puede encontrarse en [Etc05].

Tabla BD_ACTIVIDADES: **Contiene las actividades realizadas por los estudiantes en el marco de las carreras que ofrece la Facultad de Ingeniería**	
ATRIBUTO	**DESCRIPCIÓN**
ES_CI	Estudiante
MA_CODMAT	Código de materia
AS_CODAS	Asignatura
AC_FECHA	Fecha de la actividad
AC_TIPOACT	Tipo de actividad (calculado)
AC_CURRI	Indica si es curricular o no
ES_GENERACIÓN	Generación del estudiante
AC_PERIODO	Periodo en que se dicta (calculado)
AC_NOMPERIODO	Nombre del periodo (calculado)
AC_TIPOACTIVIDAD	Tipo de actividad (calculado)
AC_TIPORESULTADO	Resultado de la actividad
AC_NOTA	Nota obtenida
AC_CREDITO	Créditos obtenidos
AC_ANIO	Año en que se realiza la actividad
AC_APRUEBAS	Indica si la actividad aprueba la asignatura
AC_TIPOGEN	Tipo de generación (A=automática, C=cambio de plan, N=normal, R=revalida, V= automática a partir del cálculo
IN_CODINST	Instituto que dicta la asignatura

Tabla BD_ASIGNATURAS **Contiene las asignaturas dictadas**	
ATRIBUTO	**DESCRIPCIÓN**
CC_CODCARR	Carrera
CC_PLAN	Plan de la carrera
CC_PERFIL	Perfil de la carrera
MA_CODMAT	Materia
AS_CODAS	Asignatura
AS_NOMAS	Nombre de la asignatura
AS_CREDITOSAS	Créditos que tiene la asignatura
IN_CODINST	Código del instituto que la dicta
AS_SEMESTRE	Semestre en que se dicta

Tabla BD_CARRERAS **Contiene las carreras dictadas**	
ATRIBUTO	**DESCRIPCIÓN**
CC_CODCARR	Código de carrera (calculado)
CC_NOMCARR	Nombre de la carrera (calculado)
CC_PLAN	Plan de la carrera (calculado)
CC_PERFIL	Perfil de la carrera (calculado)
CC_NOMPERFIL	Nombre del perfil (calculado)
CC_TIPOPLAN	Tipo de plan (C=por créditos, T=tradicional)
CC_CREDITOSMIN	Cantidad mínima de créditos

Tabla BD_ESTUDIANTES **Contiene a los estudiantes que están inscriptos en carreras**	
ATRIBUTO	**DESCRIPCIÓN**
ES_CI	Cédula del estudiante
ES_NOMEST	Nombre del estudiante
ES_FECHANAC	Fecha de nacimiento del estudiante
ES_NROEST	Número de estudiante
ES_SEXO	Sexo del estudiante
ES_GENERACIÓN	Generación del estudiante, considerando la primer inscripción realizada por el estudiante a alguna carrera de la facultad.
ES_SECUNDARIA	Instituto de enseñanza secundaria de donde proviene el estudiante
LU_LUGAR	Lugar donde reside el estudiante

Tabla BD_EST_CARR **Contiene información referente a en que carreras está inscripto cada estudiantes**	
ATRIBUTO	**DESCRIPCIÓN**
ES_CI	Estudiante
CC_CODCARR	Carrera
CC_PLAN	Plan de la carrera
CC_PERFIL	Perfil de la carrera
EC_FECHAING	Fecha de ingreso a la carrera
EC_CALINSC	Calidad de la inscripción (calculado)
EC_FECHAFIN	Fecha en que finalizó sus estudios
EC_PORCAMBIO	Determina si la inscripción se realizo por cambio de plan o no
ES_GENERACIÓN	Generación del estudiante

Tabla BD_INSTITUTOS **Contiene los institutos de la Facultad de Ingenieria**	
ATRIBUTO	**DESCRIPCIÓN**
IN_CODINST	Código de instituto
IN_NOMINST	Nombre

Tabla BD_LUGARES **Contiene lugares geográficos de procedencia de los estudiantes**	
ATRIBUTO	**DESCRIPCIÓN**
LU_CODLUGAR	Código del lugar
LU_NOMLUGAR	Nombre

Tabla BD_MAT_CARR **Contiene información respecto a que materias componen cada una de las carreras**	
ATRIBUTO	**DESCRIPCIÓN**
CC_CODCARR	Carrera
CC_PLAN	Plan de la carrera
CC_PERFIL	Perfil de la carrera
MA_CODMAT	Materia
MA_CREDITOSMIN	Mínimo de créditos de la materia en la carrera

Tabla BD_MATERIAS

Contiene a las materias, las cuales componen carreras y agrupan asignaturas	
ATRIBUTO	**DESCRIPCIÓN**
MA_CODMAT	Código de la materia
MA_NOMMAT	Nombre

Tabla AX_MAPEO_ASIGNATURAS **Tabla auxiliar del sistema fuente. Permite asignar nuevos códigos a asignaturas existentes, eventualmente agrupando varias asignaturas bajo un nuevo código**	
ATRIBUTO	**DESCRIPCIÓN**
ASIGORI	Código de Asignatura original
ASIGUNI	Código unificado
NOMASIGUNI	Nombre unificado
CARR	Código de Carrera
CICLO	Código de Ciclo
CODMATUNI	Código de Materia Unificado
CODMATORI	Código de Materia Original
CREDITOS	Cantidad de Créditos
SEMESTRE	Semestre en que se dicta
CC_CODCARR	Código de Carrera en BD_CARRERAS
CC_PLAN	Código de plan en BD_CARRERAS
CC_PERFIL	Código de Perfil en BD_CARRERAS
INST	Código de Instituto

Tabla AX_MAPEO_CARRERAS **Tabla auxiliar del sistema fuente. Permite asignar nuevos códigos a carreras existentes, eventualmente agrupando varias carreras bajo un nuevo código**	
ATRIBUTO	**DESCRIPCIÓN**
CARR	Código de Carrera
CICLO	Código de Ciclo
CC_CODCARR	Código de Carrera en BD_CARRERAS
CC_NOMCARR	Nombre de la Carrera en BD_CARRERAS
CC_PLAN	Código de Plan em BD_CARRERAS
CC_PERFIL	Código de Perfil en BD_CARRERAS
CC_NOMPERFIL	Nombre Perfil en BD_CARRERAS
CC_CREDITOSMIN	Créditos mínimos de la Carreras

Tabla IN_ACTIVIDADES **Contiene las actividades realizadas por los estudiantes en el marco de las carreras que ofrece la**

Facultad de Ingeniería. Tiene información complementaria a la contenida en BD_CARRERAS	
ATRIBUTO	**DESCRIPCIÓN**
ESTCI	Cédula de identidad del estudiante
ASIG	Código de asignatura
TACT	Tipo de actividad
NOTA	Nota de la actividad (si TACT != A)
FECHA	Fecha de la actividad
CURRI	Curricular o no
TGEN	Forma de generación del registro
PER	Período de la act. (si TACT != E)
TIPOPER	Tipo del período
CODINST	Código del instituto que dicta la asignatura
OBS	Observación de la actividad

Tabla IN_CARRERAS **Contiene las carreras que ofrece la Facultad de Ingeniería. Tiene información complementaria a la contenida en BD_CARRERAS**	
ATRIBUTO	**DESCRIPCIÓN**
CARR	Código de la carrera
NOMCAR	Nombre de la carrera
PLAN	Año del plan de estudios de la carrera
CICLO	Ciclo de la carrera
NOMCIC	Nombre del ciclo
TIPOCIC	Tipo del ciclo
CRMINC	Créditos mínimos del ciclo

4.2 Medición de correctitud semántica

Las medidas de correctitud semántica se realizaron a nivel de celda. A continuación presentamos, para cada tabla fuente, las funciones aplicadas para medir la correctitud semántica.

BD_ASIGNATURAS	*CHECK_REF(as_nomas, referencial_asignaturas)*
BD_CARRERAS	*CHECK_REF(cc_nomcarr, referencial_carreras)*
BD_ESTUDIANTES	*CHECK_REF(lu_codlugar, referencial_estudiantes)*
BD_INSTITUTOS	*CHECK_REF(in_nominst,referencial_institutos)*
BD_LUGARES	*CHECK_REF(lu_nomlugar,referencial_lugares_geograf)*
BD_MATERIAS	*CHECK_REF(ma_nommat,referencial_materias)*

IN_CARRERAS	*CHECK_REF(nomcarr,referencial_carreras)*

4.3 Medición de correctitud sintáctica

Las medidas de correctitud sintáctica se realizaron a nivel de atributo. A continuación presentamos, para cada tabla fuente, las funciones aplicadas para medir la correctitud sintáctica.

BD_ACTIVIDADES	*CHECK_NULL(es_ci)*
	CHECK_RULE(ac_tiporesultado, Rango_resultados) *Rango_resultados= ('APROBADO', 'REPROBADO')*
BD_ASIGNATURAS	*CHECK_RULE(as_creditosas,Rango_creditos)* *Rango_creditos = '0 to 145'*
BD_CARRERAS	*CHECK_NULL(cc_creditosmin)*
	CHECK_RULE(cc_nomcarr,Formato_carrera) *Formato_carrera esta dado por Catalogo_carreras*
	CHECK_RULE(cc_creditosmin, Rango_creditosCarrera) *Rango_creditosCarrera= '0 to 450'*
BD_ESTUDIANTES	*CHECK_NULL(es_nroest)* *CHECK_RULE(es_ci,Formato_cedula)* *Formato_cedula ='ISNUMERIC(ES_CI)=1 AND LEN(ES_CI) > 5 AND* *(LEN(ES_CI) < 7 OR* *(LEN(ES_CI) = 7 AND SUBSTRING(CONVERT(varchar, ES_CI), 1, 1) < 7))'*
	CHECK_RULE(es_fechanac Formato_fecha) *Formato_fecha=' ISDATE(ES_FECHANAC) =1'*
	CHECK_RULE(es_nroest,Formato_nroEstudiante) *Formato_nroEstudiante='ISNUMERIC(ES_NROEST)=1 AND* *LEN(ES_NROEST) = 6'*
BD_EST_CARR	*CHECK_RULE(ec_fechaing Formato_fecha)* *Formato_fecha=' ISDATE(EC_FECHAING) =1'*
BD_INSTITUTOS	*CHECK_RULE(in_nominst,Formato_instituto)* *Formato_instituto esta dado por Catalogo_institutos*
BD_LUGARES	*CHECK_RULE(lu_nomlugar,Formato_lugar)* *Formato_lugar esta dado por Catalogo_Lugares_Geograf*
BD_MAT_CARR	*CHECK_NULL(ma_creditosmin)*
BD_MATERIAS	*CHECK_RULE(ma_nommat,Formato_materia)* *Formato_materia está dado por Catalogo_Materias*

4.4 Medición de consistencia

Tal como lo expresamos en el punto 3.3 las inconsistencias pueden medirse a diferentes niveles de granularidad, y esto depende de la naturaleza de la inconsistencia detectada. A continuación presentamos, para cada tabla fuente, las funciones aplicadas para medir inconsistencias y el nivel al cual corresponde la medición.

Tabla	***Función aplicada***	***A que se asigna el resultado de la medición***
BD_ACTIVIDADES	*CHECK_CONSTRAINT ([ac_tipoactividad], [ac_tipoactividad , ac_fecha, bd_est_carr.ec_fechaing, es_ci, bd_est_carr.es_ci], IF(bd_actividades.ac_tipoactividad<> revalida) THEN (ac_fecha>=min(bd_est_carr.ec_fechaing)) WHERE (bd_actividades.es_ci=bd_est_carr.es_ci))*	Conjunto de tablas
BD_ASIGNATURAS	*CHECK_UNIQUE (as_codas)*	atributo
BD_CARRERAS	*CHECK_UNIQUE (cc_codcarr)*	atributo
BD_ESTUDIANTES	*CHECK_FK (lu_codlugar, bd_lugares.lu_codlugar)*	Conj. de tablas
	CHECK_CONSTRAINT([es_generacion], [bd_est_carr.ec_fechaing, bd_est_carr.es_ci, es_generacion] , IF(es_generacion IS NOT NULL) THEN (es_generacion= YEAR(MIN(bd_est_carr.ec_fechaing))) WHERE (bd_estudiantes.es_ci = bd_est_carr.es_ci))	Conjunto de tablas
BD_EST_CARR	*CHECK_FK(cc_codcarr, bd_carreras.cc_codcarr)*	Celda
BD_MAT_CARR	*CHECK_CONSTRAINT ([ma_creditosmin], [bd_asignaturas. as_creditosas, ma_creditosmin, bd_asignaturas.ma_codmat, ma_codmat], IF (ma_creditosmin IS NOT NULL) THEN ma_creditosmin <= SUM(bd_asignaturas_as_creditosas) WHERE (bd_asignaturas.ma_codmat =bd_mat_carr.ma_codmat)*	Conjunto de tablas
AX_MAPEO_ASIGNATURAS	*CHECK_FK(cc_codcarr, bd_asignaturas.cc_codcarr)*	Celda
AX_MAPEO_CARRERAS	*CHECK_CONSTRAINT([cc_codcarr, cc_nomcarr], [bd_carreras.cc_nomcarr,cc_codcarr, bd_carreras.cc_codcarr, cc_plan, bd_carreras.cc_plan, cc_perfil, bd_carreras.cc_perfil], IF (cc_codcarr IS NOT NULL) THEN cc_nomcarr= bd_carreras.cc_nomcarr WHERE (ax_mapeo_carreras.cc_codcarr=bd_carreras.cc_codcarr and ax_mapeo_carreras.cc_plan = bd_carreras.cc_plan and ax_mapeo_carreras.cc_perfil= bd_carreras.cc_perfil)*	Conjunto de tablas

4.5 Medición de precisión

El nivel de precisión será medido a nivel de celda. A continuación presentamos, para cada tabla fuente, la función que aplicada para medir el nivel de precisión.

BD_LUGARES	*CHECK_LEVEL(lu_codlugar, Referencial_Nivel_Lugares_Geograf)*

La siguiente sección describe la implementación de las funciones descriptas.

5 Implementación de la medición

El mecanismo propuesto e implementado para realizar la medición de los factores de exactitud consta de tres partes: un conjunto de *metadatos* implementados en una base de datos relacional, un conjunto de *funciones de medición* y un *procedimiento principal de medición*, estos últimos implementados mediante procedimientos almacenados. El procedimiento principal de medición utiliza los metadatos para invocar a las funciones con los parámetros adecuados y almacena los resultados obtenidos como metadatos.

Este mecanismo, pese a ser desarrollado para los factores de exactitud definidos en este trabajo, fue diseñado para poder soportar la incorporación y medición de nuevos factores, así como también nuevas funciones de medición asociadas a factores ya existentes. Para incorporar un nuevo factor de calidad simplemente se debe declarar el nombre del factor en los metadatos; y para incorporar una nueva función, es necesario declarar el nombre de dicha función (y el factor asociado), implementarla e implementar su invocación en el *procedimiento principal de medición*.

5.1 Metadatos

Con el objetivo de automatizar y documentar el proceso de medición se diseñó un conjunto de metadatos que contiene información acerca de:

- los factores de calidad y tipos de errores medidos
- las funciones utilizadas para medir esos tipos de errores
- las tablas y atributos fuente sobre los que se miden los errores, las funciones de medición específicas a utilizar en cada caso y sus parámetros
- los valores de medición obtenidos.

En la figura 1 se presenta el esquema para el almacenamiento de los metadatos. El mismo se compone de tres tipos de tablas:

- Tablas para el almacenamiento de los metadatos principales. Estas tablas contienen los metadatos que definen el objeto de medición y las características particulares de cada medición (parametrización).
- Tabla de valores de exactitud, que almacena el resultado de la medición,
- Tablas auxiliares a la función de medición utilizada y particulares a cada caso, es decir los Referenciales, Catálogos, Referenciales de niveles de precisión y tablas de asignación de valores de exactitud a cada Nivel de Precisión, las cuales son utilizadas por dichas funciones como base para la medición. Su existencia y contenido dependen no sólo de la función utilizada sino también de la tabla/atributo concreto a medir del sistema fuente.

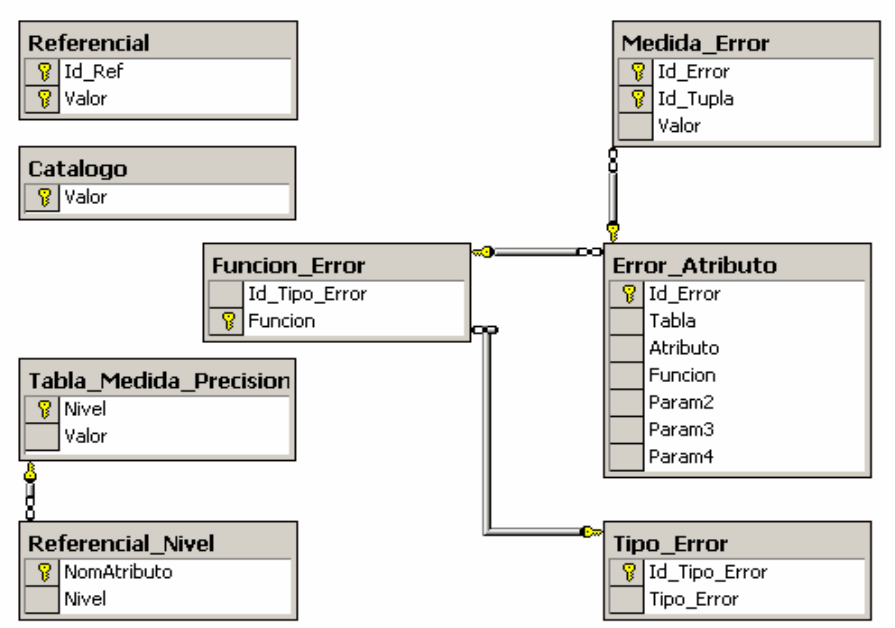


Figura 1 - Diseño lógico de los metadatos

A continuación se describen las tablas de la figura 1

5.1.1 Tablas para almacenamiento de los metadatos principales

Tabla Tipo_Error	
Atributo	**Descripción**
Id_Tipo_Error	identificador
Tipo_Error	nombre del tipo de error. Ej: Semántico

La tabla Tipo_Error representa a los tipos de error o factores considerados.

Tabla Funcion_Error	
Atributo	**Descripción**
Id_Tipo_Error	identificador del tipo de error
Funcion	nombre de la función. Ej: CHECK_NULL

La tabla Funcion_Error representa a las funciones de medición utilizadas. Cada función se utiliza para medir un único tipo de error al cual está asociada.

Tabla Error_Atributo	
Atributo	**Descripción**
Id_Error	Identificador

Tabla	Tabla del sistema fuente sobre la cual se mide el error
Atributo	Atributo de la tabla anterior sobre el cual se mide el error
Función	Nombre de función a aplicar para la medición del error (Función definida en la tabla Funcion_Error)
Param2	Parámetro auxiliar asociado a la función y cuyo significado depende de la misma. Define particularidades de la medición a realizar.
Param3	Parámetro auxiliar asociado a la función y cuyo significado depende de la misma. Define particularidades de la medición a realizar.
Param4	Parámetro auxiliar asociado a la función y cuyo significado depende de la misma. Define particularidades de la medición a realizar.

La tabla Error_Atributo modela la medición a realizar, indicando las tablas y atributo a medir del sistema fuente, el tipo de error que se mide en cada caso y las particularidades de dicha medición (básicamente la función a aplicar y su parametrización). Para cada tipo de error que se desea medir se crea una tupla en esta tabla donde se indica: tabla, atributo (si corresponde), función a utilizar y parámetros necesarios para la medición.

Para soportar en una única tabla la especificación de los parámetros necesarios para todas las funciones a utilizar en la medición, la semántica de los atributos *Param2, Param3 y Param4* está dada por la función de medición utilizada. Estos atributos almacenan los parámetros necesarios para realizar la medición definida en la sección 4.

La granularidad de la medición, que puede ser a nivel de celda, tupla, atributo o conjunto de tablas, también se indica en esta tabla. Si el atributo *Atributo* es nulo el error se mide a nivel de tupla, mientras que si se indica un atributo específico de la tabla, la granularidad es a nivel de celda.

A continuación se detalla la semántica de los atributos *Param2, Param3 y Param4* en función de cada una de las funciones de cálculo.

Función: CHECK_REF	
Factor de calidad: Correctitud Semántica	
Param2:	Nombre de la tabla referencial utilizada como fuente confiable: Ej: Referencial_Carreras
Param3:	Expresión generadora del identificador de tupla, el cual se compara con el identificador del referencial. Se utilizó la función CHECKSUM como generadora de identificadores de tupla Ej: CHECKSUM(CAST(SUBSTRING(CC_PLAN, 18, 2) AS int),

Función: CHECK_RULE	
Factor de calidad: Correctitud Sintáctica	
Param3:	Determina como se implementó la regla a chequear. Los valores posibles para este atributo son: 'CATALOGO', 'SQL' o 'PROCEDURE'
Param2:	Según el valor del parámetro Param3, este parámetro se interpreta como: • Nombre de la tabla utilizada como catálogo de valores sintácticamente correctos (CATALOGO) • Condición expresada en SQL que implementa la regla (SQL) • Nombre del procedimiento almacenado que implementa el chequeo de la regla (PROCEDURE)

Función: CHECK_NULL

Factor de calidad: Correctitud Sintáctica	
Param2:	No utiliza este parámetro auxiliar.
Param3:	No utiliza este parámetro auxiliar.

Funcion: CHECK_UNIQUE	
Factor de calidad: Consistencia	
Param2:	No utiliza este parámetro auxiliar.
Param3:	No utiliza este parámetro auxiliar.

Funcion: CHECK_FK	
Factor de calidad: Consistencia	
Param2:	Tabla contra la que se chequea la restricción de clave foránea
Param3:	Atributo de la tabla indicada en Param2, contra el que se chequea la restricción de clave foránea

Funcion: CHECK_CONSTRAINT	
Factor de calidad: Consistencia	
Param4	Determina como se implementó la restricción a chequear Los valores posibles para este atributo son: 'SQL' o 'PROCEDURE'
Param2:	Según el valor del parámetro Param4, este parámetro se interpreta como:

Param4	**Significado**
SQL	Precondición para que valga la restricción
PROCEDURE	Nombre del procedimiento almacenado que implementa la restricción

Param3: Según el valor del parámetro Param4, este parámetro se interpreta como:

Param4	**Significado**
SQL	Restricción expresada en SQL
PROCEDURE	No utiliza este parámetro auxiliar.

Cabe señalar que, en caso de no cumplirse la precondición que habilita el chequeo de la restricción, el dato se considera correcto y como consecuencia, pueden aparecer inconsistencias, pues el dato aparece como correcto cuando no lo es.

Ilustraremos esta situación con un ejemplo. Supongamos que se desea chequear un atributo numérico, considerándolo correcto si presenta valores menores que 5. Para realizar este chequeo de consistencia es necesario que el valor del atributo cumpla dos precondiciones: que el atributo no sea nulo y que efectivamente tenga un valor numérico. Si se viola alguna de estas precondiciones el chequeo de consistencia en cuestión no tiene sentido, y la función CHECK_CONSTRAINT devuelve como resultado que el valor del atributo cumple la condición.

Desde un punto de vista más general, si efectivamente se espera que ese atributo tenga valores menores que 5, los casos en que el atributo es nulo o no numérico son casos de error y por lo tanto deben medirse explícitamente mediante CHECK_NULL y CHECK_CONSTRAINT.

Es imprescindible diseñar correctamente las mediciones a realizar, de lo contrario los resultados no reflejarán lo esperado y pueden presentarse inconsistencias.

<table>
<tr><td colspan="2">Funcion: CHECK_LEVEL</td></tr>
<tr><td colspan="2">Factor de calidad: Precisión</td></tr>
<tr><td>Param4</td><td>Determina como se calcula el nivel de precisión
Los valores posibles para este atributo son: 'SQL' o 'TABLA'</td></tr>
<tr><td>Param2:</td><td>Según el valor del parámetro Param4, este parámetro se interpreta como:
<table>
<tr><th>Param4</th><th>Significado</th></tr>
<tr><td>SQL</td><td>Expresión en SQL que determina el nivel de precisión para el valor dado</td></tr>
<tr><td>TABLA</td><td>Nombre de la tabla que asigna a cada valor posible del atributo un nivel de precisión</td></tr>
</table></td></tr>
<tr><td></td><td></td></tr>
<tr><td>Param3:</td><td>Nombre de la tabla tipo Tabla_Medida_Precision que asigna a cada nivel de precisión un valor de EXACTITUD con respecto a la precisión esperada</td></tr>
</table>

Esta función calcula el nivel de precisión del dato y a partir de éste asigna el correspondiente valor de exactitud establecido en la tabla Tabla_Medida_Precision definida. Dicha tabla debe mantener la relación entre el nivel de precisión y el valor de exactitud asociado; una posibilidad es definir el valor de exactitud de forma que refleje la distancia entre la precisión encontrada y la esperada.

5.1.2 Tabla de Valores de exactitud resultantes de la medición

Tabla Medida_Error	
Atributo	**Descripción**
Id_Error	Identificador del error medido definido en la tabla Error_Atributo
Id_Tupla	Identificador de la tupla sobre la cual se mide el error
Valor	Resultado de la medición, en este caso un valor de exactitud entre 0 y 1

La tabla Medida_Error almacena los resultados de las mediciones especificadas en la tabla Error_Atributo. Para cada tupla de la *Tabla* o *Tabla/Atributo* objeto de medición, según lo definido en la tabla *Error_Atributo*, la tabla Medida_Error contiene una tupla con un valor de exactitud asociado al error medido.

Para poder almacenar todos los valores de exactitud calculados en una única tabla fue necesario idear un mecanismo que generara un identificador único para cada tupla e independiente de la tabla y su identificador particular. La forma de generación de dicho identificador depende del RDBMS utilizado, por ejemplo en el caso de PostgreSQL podría utilizarse el UID u en Oracle el *rowid*, los cuales son números que identifican unívocamente a cada tupla de la base de datos.

En este caso la solución se implementó sobre Microsoft SQLServer, el cual en su versión 2000 no cuenta con una funcionalidad similar a la de *rowid* de Oracle. Se investigaron posibles soluciones, no encontrando ninguna implementación documentada que garantizase la unicidad de los valores obtenidos. Pese a esto se optó por aplicar la función CHECKSUM sobre los valores de cada tupla, emulando el identificador de tupla de esta forma y sabiendo que este identificador no garantizaba unicidad. Este y otros inconvenientes encontrados durante la implementación se analizan en la sección 6.

5.1.3 Tablas auxiliares

Referenciales

Las tablas referenciales son utilizadas en la implementación de la función CHECK_REF. Las mismas simulan la realidad o una fuente confiable contra la cual comparar los datos del sistema fuente.

Se define una tabla referencial por cada tupla de la tabla Error_Atributo que utilice esta función, es decir, para cada atributo de cada tabla sobre el cual se medirá la correctitud semántica. Cada referencial contiene, para cada instancia válida, el identificador de dicha instancia y el valor correcto/real del atributo sobre el que se desea medir la correctitud semántica.

Atributo	**Descripción**
Id_Ref	Identificador de cada instancia de la realidad. (En este caso de genera aplicando la función CHECKSUM sobre los atributos clave)
Valor	Valor correcto/real del atributo a validar

Para este caso de estudio se definieron y cargaron las siguientes tablas referenciales:

- *Referencial_Carreras*
- *Referencial_Asignaturas*
- *Referencial_Estudiantes*
- *Referencial_Institutos*
- *Referencial_Lugares_Geograf*
- *Referencial_Materias*

Catálogos

Los catálogos son utilizados en la implementación de la función CHECK_RULE. Los mismos representan diccionarios de valores sintácticamente correctos, contra los cuales se chequea la correctitud sintáctica de los datos de un sistema fuente.

Se define un catálogo por cada dominio a chequear sintácticamente, por lo tanto cada catálogo puede ser utilizado en el cálculo por varias tuplas de la tabla Error_Atributo que utilicen la función CHECK RULE.

Cada catálogo contiene un conjunto de valores sintácticamente correctos.

Atributo	**Descripción**
Valor	Valor sintácticamente correcto para el dominio en cuestión.

Para este caso de estudio se definieron y cargaron los siguientes catálogos:

- *Catalogo_Carreras*
- *Catalogo_Institutos*
- *Catalogo_Lugares_Geograf*
- *Catalogo_Materias*

Referenciales de nivel de precisión

Las tablas Referenciales de nivel de precisión definen un nivel de precisión para cada valor posible de un atributo dado. Estas tablas son utilizadas en la implementación de la función CHECK_LEVEL para determinar el nivel de precisión de los valores de los atributos. A partir de la determinación de dicho nivel, se asignará el valor de exactitud que corresponda según lo definido en la tabla *Tabla_Medida_Precision* que se describirá más adelante.

Se define una tabla referencial de nivel por cada atributo sobre el cual se medirá la precisión.

Atributo	**Descripción**
NomAtributo	Valor del atributo NomAtributo de la tabla en cuestión.
Nivel	Valor numérico asignado al valor del atributo.

Para este caso de estudio se definieron y cargaron los siguientes Referenciales de Nivel de precisión:

- *Referencial_Nivel_Lugares_Geograf*

Tabla_Medida_Precisión

Las tablas Tabla_Medida_Precision son utilizadas en la implementación de la función CHECK_LEVEL y definen la correspondencia entre los distintos niveles de precisión y el valor de exactitud asignado.

Se define una tabla medida-precision por cada dominio sobre el cual interesa medir la precisión.

Atributo	**Descripción**
Nivel	Nivel de precisión en la clasificación definida.
Valor	Valor de exactitud asignado al nivel (valor numérico entre 0 y 1)

Para este caso de estudio se definió y cargó únicamente la siguiente Tabla_Medida_Precision:

- *Tabla_Medida_Precision_Lugares_Geograf*

5.2 Funciones y procedimientos

La medición de la *exactitud* de los datos fuente se realiza a partir de los datos cargados en la metadata presentada anteriormente, y se almacena en una de sus tablas: *Medida_Error*.

Dicho cálculo es realizado con el procedimiento almacenado *load_table*, el cual básicamente recorre la tabla *Error_Atributo* y obtiene para cada error ingresado, sus datos asociados, entre ellos:

- *Tabla* y *Atributo* del sistema fuente sobre el cual se aplica el error
- *Función* a aplicar
- Parámetros particulares de dicha función.

A partir de los datos anteriores, recorre la tabla especificada del sistema fuente y aplica a cada una de sus tuplas el cálculo correspondiente a la función indicada según los parámetros obtenidos, insertando finalmente el resultado en la tabla *Medida_Error*.

La idea inicial era independizar este procedimiento de la función especificada, haciéndolo paramétrico e implementando cada función externamente al procedimiento. De esta forma se lograría que la incorporación de una nueva función no afecte el código del procedimiento. Debido a restricciones en el DBMS utilizado y a razones de performance, se abandonó temporalmente dicha iniciativa.

A continuación se presenta el pseudocódigo del procedimiento *load_table* (un pseudocódigo más detallado se encuentra en el Anexo I):

- Para cada error ingresado en la tabla ERROR_ATRIBUTO
  - Obtiene los datos *Id_Error*, *Tabla*, *Atributo*, *Funcion* y Parámetros (*Param2, Param3, Param4*)
  - Para cada tupla de *Tabla*
    - Si la *Funcion* es **CHECK_NULL**
      - Chequea si el valor de *Atributo* es nulo en cada tupla de *Tabla*
    - Sino, Si la *Funcion* es **CHECK_REF**
      - Chequea si el valor de *Atributo* coincide con el valor correspondiente en el Referencial.
    - Sino, Si la *Funcion* es **CHECK_FK**
      - Para cada tupla de *Tabla* chequea la existencia del valor de *Atributo* en el *Atributo Destino* (se obtiene de *Param2*) de alguna tupla de Tabla Destino (se obtiene de Param3).
    - Sino, Si la *Funcion* es **CHECK_UNIQUE**
      - Para cada tupla de *Tabla* chequea si el valor de *Atributo* es único en esa tabla.
    - Sino, Si la *Funcion* es **CHECK_RULE**
      - Si el tipo de cálculo (en *Param3*) es 'SQL'
        - Verifica si la tupla cumple la sentencia SQL (en *Param2*) dada
      - Sino, Si el tipo de cálculo (en *Param3*) es 'CATALOGO'
        - Verifica si el valor de *Atributo* existe en el Catálogo (en *Param2*)
      - Sino, Si el tipo de cálculo (en *Param3*) es 'PROCEDURE'
        - Verifica - ejecutando el procedimiento (*Param2*) – si la tupla cumple o no la regla esperada.
    - Sino, Si la *Funcion* es **CHECK_CONSTRAINT**
      - Si el tipo de cálculo (en *Param4*) es 'SQL'
        - Verifica si la tupla cumple la condición del *IF (Param2)*, y si la cumple, verifica la condición del *THEN (Param3)*, aplicando las consultas SQL correspondientes.
      - Sino, Si el tipo de cálculo (en *Param4*) es 'PROCEDURE'
        - Verifica - ejecutando el procedimiento (*Param2*) – si la tupla cumple o no la condición esperada.
    - Sino, Si la *Funcion* es **CHECK_LEVEL**
      - Si el tipo de cálculo (en *Param4*) es 'TABLA'

    - Obtiene el nivel de precisión de *Atributo*, consultando las tablas 'Referencial de Nivel' (*Param2*) y 'Nivel Precision' (*Param3*).
  - Sino, Si el tipo de cálculo (en *Param4*) es 'SQL'
    - Calcula el Nivel de jerarquía correspondiente al valor de *Atributo*, aplicando la sentencia SQL (en *Param2*)
    - A partir del Nivel de jerarquía anterior, obtiene - de la tabla 'Nivel Precision' (*Param3*) - el valor de precisión a asignar
- Inserta el resultado del chequeo de la tupla en *Medida_Error*

5.3 Dificultades encontradas y definiciones realizadas durante la medición

Durante el desarrollo del presente estudio se encontraron diversas dificultades a partir de las cuales, luego de su análisis y evaluación de alternativas, se realizaron algunas definiciones o supuestos.

Se definió la granularidad y los valores posibles de exactitud en cada tipo de error a medir. Se encontraron problemas para identificar las tuplas de la metadata independientemente de las claves. Se encontraron problemas propios de DBMS utilizado y la integración de datos provenientes de distintos DBMSs. Se discutieron y definieron criterios para la medición mediante las funciones *Check Rule* y *Check Constraint*, y se discutió la posible dependencia entre los diferentes chequeos.

Todas estas discusiones y definiciones se encuentran explicadas en detalle en el Anexo II.

6 Resultados obtenidos

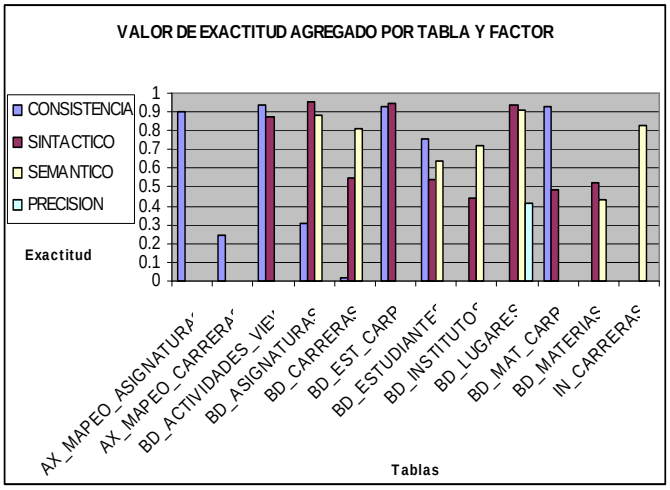
En todos los casos las medidas se tomaron al nivel de granularidad más bajo posible para cada tipo de error o factor. (EJ: el nivel más bajo de correctitud sintáctica es el de celda, mientras que para la consistencia respecto de una restricción de clave foránea es el conjunto de las tablas participantes en la restricción).

Fue preciso diseñar mecanismos de agregación para obtener, para cada factor medido, medidas en niveles de granularidad más altos. Por ejemplo, se consideró que una tupla es correcta según un factor si todas las medidas de dicho factor realizadas sobre las celdas de la tupla son correctas, de lo contrario se la considera incorrecta. Para obtener medidas a nivel de tabla se realizó el promedio de las medidas a nivel de tupla. La Tabla 2 muestra los valores obtenidos para cada tabla y factor de calidad. Las tablas con los resultados de exactitud a un mayor nivel de detalle se encuentran en el Anexo III.

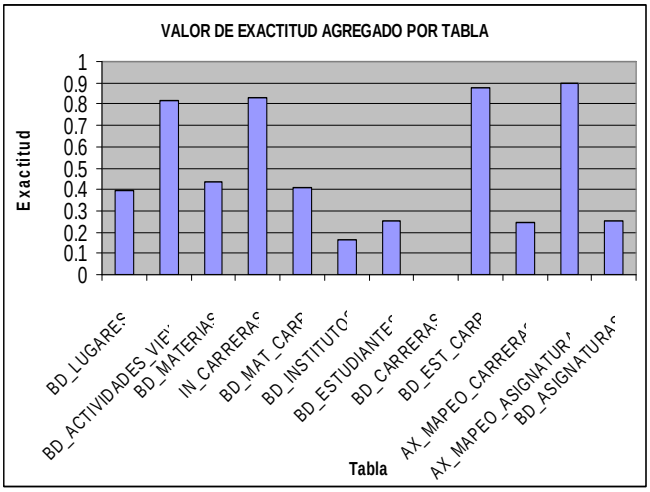
Tabla	Consistencia	Sintáctico	Semántico	Precisión
AX_MAPEO_ASIGNATURAS	0.897268			
AX_MAPEO_CARRERAS	0.245283			
BD_ACTIVIDADES_VIEW	0.938552	0.874075		
BD_ASIGNATURAS	0.308788	0.9519	0.887173	
BD_CARRERAS	0.021276	0.553191	0.80851	
BD_EST_CARR	0.931623	0.943443		
BD_ESTUDIANTES	0.752688	0.540434	0.636556	
BD_INSTITUTOS		0.444444	0.722222	
BD_LUGARES		0.939393	0.90909	0.416666
BD_MAT_CARR	0.924302	0.482071		
BD_MATERIAS		0.521912	0.434262	
IN_CARRERAS			0.829787	

Tabla 2– Valores de exactitud agregados por Tabla y Factor de Calidad

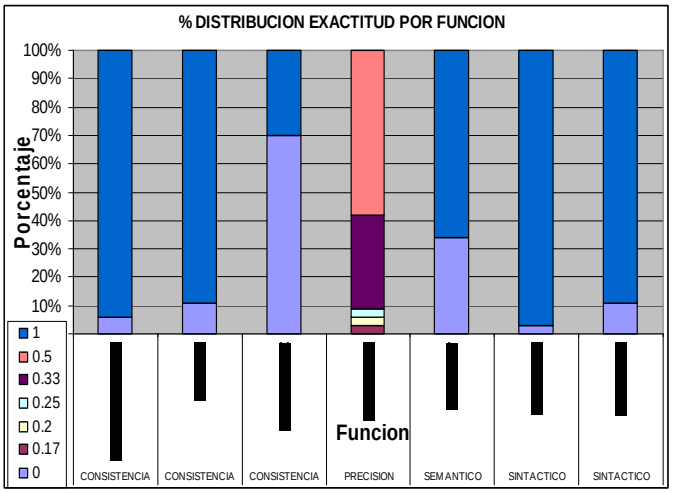
A continuación se resumen los resultados obtenidos. La Grafica 1 muestra los valores de exactitud agregados para cada tabla y factor de calidad (corresponde con la Tabla 2). La Grafica 2 indica el porcentaje de tuplas de cada tabla que no tiene ningún tipo de error (para ninguno de los factores considerados).



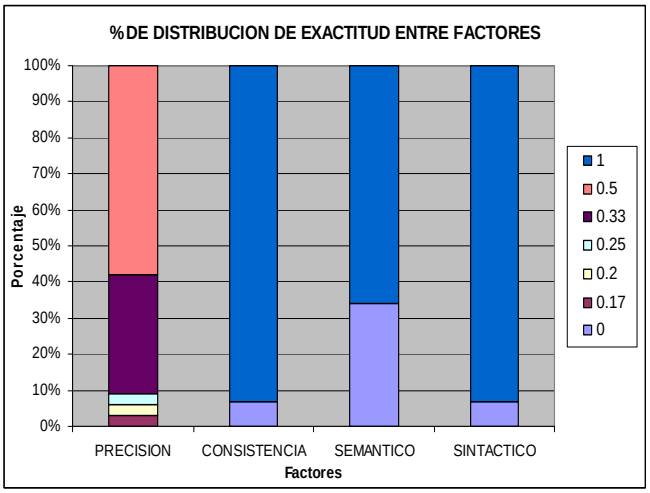
Gráfica 1– Valor de exactitud agregado por Tabla y Factor



Gráfica 2– Valores de exactitud agregado por Tabla



Gráfica 3–Distribución de los valores de exactitud en las Funciones utilizadas



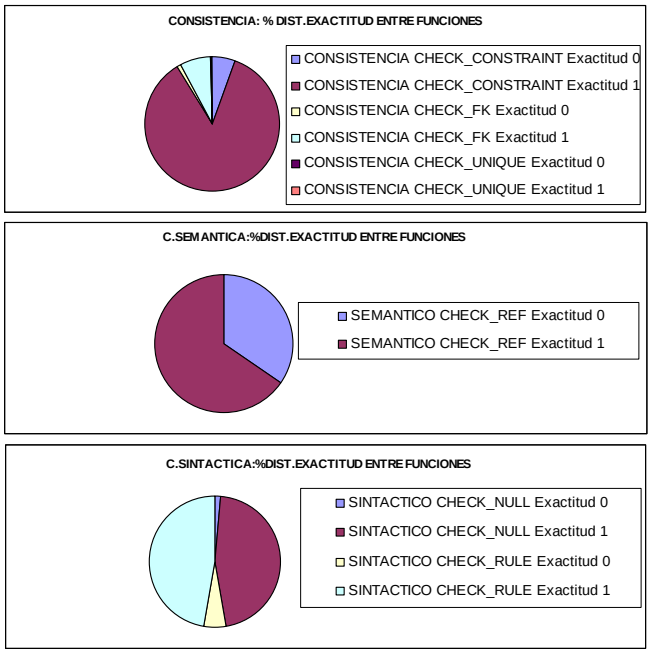
Gráfica 4 – Distribución de los valores de exactitud en los Factores considerados

La gráfica 3 muestra porcentajes de exactitud independientes para cada función, y por lo tanto no indica la contribución de cada función a la exactitud del factor de calidad a la cual pertenece. La gráfica 5 muestra dicha contribución dividiendo en tres gráficas, una para cada factor considerado.

Analizando la información obtenida llegamos a conclusiones sobre la exactitud en las fuentes de datos, de las cuales presentamos lo más relevante:

El factor de calidad más afectado (Gráfica 4) es *Correctitud Semántica* en la función *CHECK_REF*, luego siguen los factores Correctitud Sintáctica y Consistencia, pero porcentualmente en menor medida. De estos últimos, el que exhibe menor exactitud es el resultado de la función *CHECK_UNIQUE* (Gráfica 3), sin embargo quien contribuye porcentualmente en mayor medida a la ausencia de exactitud en el factor Consistencia es la función *CHECK_CONSTRAINT* (Gráfica 5) a pesar de obtenerse una alta exactitud en dicha función (Esto es debido a que esta función abarcó a una cantidad de tuplas mucho mayor que el resto de las funciones) y *CHECK_RULE* para el caso de Correctitud Sintáctica.

Bajando de nivel estos resultados (Tabla 4 del Anexo III), podemos observar que:



Gráfica 5 – Distribución de la exactitud de cada factor entre sus funciones

- **Factor Correctitud Semántica**: En la función CHECK_REF aplicada a la tabla BD_MATERIAS es donde se obtiene menor exactitud (atributo MA_NOMMAT), pero si consideramos su contribución a la exactitud del Factor (lo cual pesa en la agregación a nivel de factor) BD_ESTUDIANTES es la mas afectada (atributo LU_CODLUGAR)
- **Factor Consistencia:** La Función CHECK_UNIQUE figura con resultados de menor exactitud, en particular la tabla BD_CARRERAS es la más afectada (atributo CC_CODCARR), siguiendo la tabla BD_ASIGNATURAS (atributo AS_CODAS). Sin embargo es claramente más importante la contribución de la función CHECK_CONSTRAINT a la falta de exactitud del Factor, en particular la tabla BD_ACTIVIDADES (atributo AC_TIPOACTIVIDAD), seguido luego por la Función CHECK_FK en las tablas BD_EST_CARR (atributo CC_CODCARR) y BD_ESTUDIANTES (atributo LU_CODLUGAR)
- **Factor Correctitud Sintactica**, Aunque las tablas con menor valor de exactitud son BD_INSTITUTOS, BD_MATERIAS (ambas en la función CHECK_RULE) y BD_MAT_CARR (función CHECK_NULL) , se destaca la cantidad de registros afectados en las funciones CHECK_RULE y CHECK_NULL en la tabla BD_ACTIVIDADES (atributos AC_TIPORESULTADO y ES_CI respectivamente) y la función CHECK_RULE en la tabla BD_ESTUDIANTES (atributos ES_CI y ES_NROEST)

Por otro lado, analizando la Tabla 2, las tablas mas afectadas en su exactitud en cada factor son:

- **Factor Correctitud Semantica**: BD_MATERIAS y BD_ESTUDIANTES
- **Factor Consistencia**: BD_CARRERAS, BD_ASIGNATURAS, AX_MAPEO_CARRERAS
- **Factor Correctitud Sintactica**: BD_INSTITUTOS, BD_MAT_CARR, BD_MATERIAS, BD_ESTUDIANTES

A su vez, de la Gráfica 3 podemos obtener las tablas con menos porcentaje de registros sin error son BD_CARRERAS, BD_INSTITUTOS, BD_ESTUDIANTES, BD_ASIGNATURAS, AX_MAPEO_CARRERAS

El objetivo podría ser concentrarse en la exactitud de determinadas fuentes (las que están asociadas a algún proceso particular o simplemente las más afectadas) o concentrarse en las mediciones cuya falta de exactitud más afecta al factor considerado.

Consideramos ambos objetivos simultáneamente y nos concentramos en las tabla/atributo más importantes buscando identificar el problema que determina su bajo valor de exactitud:

Tabla	Función	Atributo	Problema identificado
BD_MATERIAS	CHECK_REF	MA_NOMMAT	Se identificó la misma materia con distinto nombre, así como distintos criterios de nombrado. Todos nombres no identificados formalmente como materias.
BD_ESTUDIANTES	CHECK_REF	LU_CODLUGAR	La mitad de los errores son consecuencia de un error Sintactico: Error en la Cedula del Estudiante (ES_CI), valor no existente o mal formadas ; el resto se corresponde a lugares (LU_CODLUGAR) fuera de rango
BD_CARRERAS	CHECK_UNIQUE	CC_CODCARR	Se repiten los códigos de carrera, el problema identificado es que el código de la carrera es único para un nombre de carrera, pero la carrera se repite para distintos planes y ciclos. La unicidad se mantiene en la tripleta (plan,carrera,ciclo). Se debe modificar el chequeo realizado.
BD_ASIGNATURAS	CHECK_UNIQUE	AS_CODAS	Se encontraron asignaturas que se repiten para distintas materias. El código de la asignatura en esta tabla es único para cada materia, por lo cual se debe modificar el chequeo realizado.
BD_ACTIVIDADES	CHECK_CONSTRAINT	AC_TIPOACTIVIDAD	Se constató que figuran materias aprobadas con fecha anterior al ingreso del estudiante a la facultad, por lo tanto hay una incoherencia en los datos ingresados. No se identificó el origen, pero son cerca de 1000 estudiantes afectados y 300 fechas de ingreso. Es posible que se corresponda con algún update realizado directamente en la Base de datos.
BD_INSTITUTOS	CHECK_RULE	IN_NOMINST	Se encontraron diversos nombres de institutos y abreviaturas no consideradas en el Catalogo contra el cual se realiza la comparación de correctitud. Sería necesario actualizar el Catálogo y quizas también revisar los valores posibles en la aplicación.
BD_MATERIAS	CHECK_RULE	MA_NOMMAT	Existen valores de MA_NOMMAT que no se corresponde con nombres válidos de materias (Ej: PRIMER SEMESTRE, QUINTO AÑO). Probablemente sea debido a un error conceptual en el ingreso de los datos por parte de algún usuario.
BD_MAT_CARR	CHECK_NULL	MA_CREDITOSMIN	Se identificó gran cantidad de valores nulos, se desconoce el origen, pero siguen cierto patrón dado por el plan y el perfil
BD_ACTIVIDADES	CHECK_RULE	AC_TIPOACTIVIDAD	Se encontraron valores inválidos de AC_TIPOACTIVIDAD, son abreviaturas de lois dos valores posibles y es debido al la falta de control en el ingreso de estos datos a través de la aplicación.
BD_ACTIVIDADES	CHECK_NULL	ES_CI	Existen valores nulos en la cedula (también valores fuera de rango), probablemente por error en la aplicación o al ingresar los datos
BD_ESTUDIANTES	CHECK_NULL	ES_CI	Existen valores nulos en la cedula, probablemente por error en la aplicación o al ingresar los datos

Tabla	Función	Atributo	Problema identificado
BD_ESTUDIANTES	CHECK_RULE	ES_CI	Se confirman los errores en la Cedula del Estudiante (ES_CI), valor no existente o mal formadas
BD_ESTUDIANTES	CHECK_RULE	ES_NROEST	Se identificaron valores no numéricos, valores nulos y valores fuera de rango. Se desconoce el origen del problema, podria ser un problema en la aplicación o un error al realizar modificaciones directas a la base de datos (dado que siguen un patrón claro)

7 Conclusiones

En este trabajo se realizó una experiencia de medición de factores de calidad de datos en bases de datos fuentes del sistema de DW de Enseñanza de la Facultad de Ingeniería.

Este trabajo resulta de mucho valor para nuestro grupo de investigación ya que es la primera experiencia que tiene dicho grupo en la aplicación a un caso real, de definiciones y técnicas teóricas sobre medición de factores de calidad en sistemas multi-fuente.

Se decidió trabajar con la dimensión de calidad: exactitud de datos. En el contexto de dicha dimensión se definieron los factores de calidad que resultarían de interés medir en este caso particular (correctitud sintáctica, correctitud semántica, consistencia y precisión), y para dichos factores se definieron los procedimientos de medición adecuados a los distintos tipos de error considerados. También se decidió qué datos de cada tabla fuente serían medidos. Para tomar estas decisiones se debió, por un lado, profundizar en los conceptos teóricos existentes en la literatura sobre el factor de calidad exactitud, y por otro lado, lograr un conocimiento minucioso sobre los datos fuentes que se querían medir.

El trabajo realizado permitió reafirmar algunos aspectos previamente considerados, tales como:

- La importancia de contar con una apropiada Granularidad en la medición, adecuada para viabilizar el trabajo posterior sobre los datos recolectados.
- La necesidad de una metadata de soporte a la medición y la importancia práctica de contar con un procedimiento que automatice dicha medición.
- Lo fundamental que resulta el conocimiento del dominio de los datos para determinar de forma acertada qué elementos medir (y cómo), de manera de obtener buenos resultados.
- La valiosa información obtenida, al evidenciar el origen de la falta de calidad en ciertos datos y permitir predecir su impacto en la propagación de datos y prevenirlo; ya sea corrigiendo el origen del problema, optando por una fuente de datos alternativa, etc.

A su vez, posibilitó extender el conocimiento teórico existente y contrastarlo con un caso real, al concretar la implementación del marco de trabajo necesario para la medición de un factor de calidad y tomar contacto con las consideraciones y dificultades que surgen en la práctica. Como resultado, podemos mencionar los siguientes aspectos relevantes:

- Se identificaron varias métricas asociadas a un mismo factor, las cuales fueron implementadas mediante distintas funciones. La lista de funciones presentada no es exhaustiva, sólo se identificaron las que se consideraron apropiadas al caso particular.
- Surgió el problema de la combinación de valores de exactitud a distintos niveles, por ejemplo la combinación en las distintas funciones para obtener un valor único a nivel de factor; a nivel de tupla, agregando los distintos valores de exactitud de los distintos atributos; a nivel de tabla agregando los valores obtenidos a nivel de tupla, y finalmente la combinación de los valores obtenidos a partir de los distintos factores.

- Encontramos que sutiles diferencias en la medición implementada puede provocar resultados de exactitud marcadamente diferentes. Por ejemplo, el no considerar los posibles errores en los datos, en una regla utilizada en cierta medición, puede ocasionar que se propaguen inadvertidamente dichos errores a la métrica. Por tal razón, es importante determinar cuidadosamente si un determinado error debe impactar o no cierta métrica, y en consecuencia definir la función a utilizar y su parametrización. Por otro lado, se entendió muy relevante la coherencia y completitud de las medidas y fuentes seleccionadas. Es decir, las medidas tomadas sobre las distintas fuentes no deberían considerarse aisladamente, sino en conjunto, dado que los chequeos realizados no son independientes entre sí (coherencia), y a su vez, el subconjunto de fuentes elegidas y las medidas tomadas entre ellas deben proveer los datos necesarios para la propagación a realizar (completitud), de lo contrario el resultado puede verse afectado por la falta de una medida en alguna de las fuentes.

Se cumplió con los objetivos principales propuestos de experimentar los conceptos teóricos sobre medición de factores de calidad en casos reales determinando así la calidad de los datos fuentes en un sistema de información multi-fuente concreto, y obtener la información que servirá de base para el siguiente estudio de propagación del factor de calidad exactitud. El próximo trabajo versará básicamente sobre la propagación de los valores de exactitud de las fuentes hasta el usuario final, estudiando sus particularidades y comparando el resultado concreto obtenido con el valor real medido.

.

8 Referencias

[Bal98] Ballou, D.; Wang, R.; Pazer, H.; Tayi, G.: "*Modelling Information Manufacturing Systems to Determine Information Product Quality*". Management Science, Vol. 44 (4), April 1998.

[Bob98] Bobrowski, M.; Marré, M.; Yankelevich, D.: "A Software Engineering View of Data Quality". 2nd Int. Software Quality Week Europe (QWE'98), Brussels, Belgium, 1998.

[Bou02] Bouzeghoub, M.; Kedad, Z.: "Quality in Data Warehousing". Information and database quality, Piattini, M.; Calero, C.; Genero, M. (eds), Kluwer Academic Publisher, 2002.

[Etc03] Etcheverry, L.; Marrero, P.: "Sistema de Data Warehouse de Enseñanza en la Facultad de Ingeniería". Trabajo de grado (tutor: Adriana Marotta), Facultad de Ingeniería, Universidad de la República, Uruguay, 2003.

[Etc05] Etcheverry, L.; Gatto, P.; Tercia, S.; Marotta, A.; Peralta, V.: "Análisis del proceso de carga del Sistema de Data Warehousing de Enseñanza de la Facultad de Ingeniería". Reporte técnico, In.Co., Universidad de la República, Uruguay, diciembre 2005.

[Jar97] Jarke, M.; Vassiliou, Y.: "*Data Warehouse Quality: A Review of the DWQ Project*". In Proc. 2nd Conference on Information Quality (IQ'1997), Cambridge, USA, 1997.

[Lab05] Laboisse, B.: "*BDQS, une approche dans la mesure de la qualité de données d'un CRM : principes de base (les attributs de la qualité), intégration des outils métier de marketing direct dans la mesure*". Séminaire CRM & Qualité des Données, Paris, France, 2005.

[Mec02] Mecella, M.; Scannapieco, M.; Virgillito, A.; Baldoni, R.; Catarci, T.; Batini, C.: "Managing Data Quality in Cooperative Information Systems". In Proc. on the Confederated Int. Conf. DOA, CoopIS and ODBASE (DOA/CoopIS/ODBASE'02), Irvine, USA, 2002.

[Nau99] Naumann, F.; Leser, U.; Freytag, J.C.: "Quality-driven Integration of Heterogeneous Information Systems". In Proc. of the 25th Int. Conf. on Very Large Databases (VLDB'99), Scotland, 1999.

[Nau00] Naumann, F.; Rolker, C.: "*Assessment Methods for Information Quality Criteria*". In Proc. of the MIT Conf. on Information Quality (IQ'00), Cambridge, USA, 2000.

[Per05] Peralta, V.: "A Framework for Analysis of Data Accuracy". Internal Report, In.Co., Universidad de la República, Uruguay, 2005.

[Per06] Peralta, V.: "Data Freshness and Data Accuracy: a State of the Art". Technical Report TR13-06, In.Co., Universidad de la República, Uruguay, Marzo de 2006.

[Pip02] Pipino, L.L.; Lee, Y.W.; Wang, R.: "Data Quality Assessment". Communications of the ACM, vol. 45, No. 4ve, April 2002.

[Red96] Redman, T.: "Data Quality for the Information Age". Artech House, 1996.

[Ter05] Tercia, S.; Peralta, V.: "Análisis de la Exactitud de los Datos: Factores y Métricas". Internal Report, In.Co., Universidad de la República, Uruguay, 2005.

[Ter05a] Tercia, S.; Gatto, P.: "Propagación de valores de correctitud a través de operadores de álgebra relacional". Reporte interno, In.Co., Universidad de la República, Uruguay, 2005.

[Wan96] Wang, R.; Strong, D.: "Beyond accuracy: What data quality means to data consumers". Journal on Management of Information Systems, Vol. 12 (4):5-34, 1996.

Anexo I – Procedimiento almacenado de medición

A continuación se presenta el pseudocódigo detallado del procedimiento almacenado que implementa la medición de la exactitud (*load_table):*

- Se setea el sistema fuente y el que contiene las tablas auxiliares (el sistema que contiene la metadata es el local)
- Para cada error ingresado en la tabla ERROR_ATRIBUTO
    - Obtiene los datos *Id_Error*, *Tabla*, *Atributo*, *Funcion* y Parámetros (*Param2*, *Param3*, *Param4*)
    - Si la *Funcion* es **CHECK_NULL**
        - Para cada tupla en la tabla *Tabla*
            - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
            - Verifica si el atributo *Atributo* es NULO
            - *(Si el atributo es NULO el resultado es 0, de lo contrario 1)*
            - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación de NULO en la tabla *Medida_Error*
    - Sino, Si la *Funcion* es **CHECK_REF**
        - Setea el Referencial a partir del parámetro *Param2*
        - Setea el algoritmo para calcular el identificador de la instancia del Referencial, a partir del parámetro *Param3*
        - Para cada tupla en la tabla *Tabla*
            - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
            - Calcula el identificador de la instancia a partir de la tupla y verifica si se corresponde con el identificador de alguna instancia del Referencial y si coincide el valor del *Atributo*.
            - *(Si la instancia y el valor del Atributo coinciden con el contenido en el Referencial, el resultado es 1, de lo contrario 0)*
            - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
    - Sino, Si la *Funcion* es **CHECK_FK**
        - Setea la Tabla Destino (es decir la referenciada por la Foreign Key) a partir del parámetro *Param2*
        - Para cada tupla en la tabla *Tabla*
            - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
            - Setea el nombre del atributo destino (es decir el referenciado por la Foreign Key) a partir del parámetro *Param3*
            - Verifica si el valor de *Atributo* existe en el campo *Atributo Destino* de alguna tupla de la *Tabla Destino*
            - *(Si el valor de Atributo existe en la Tabla Destino, el resultado es 1, de lo contrario 0)*
            - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
    - Sino, Si la *Funcion* es **CHECK_UNIQUE**
        - Para cada tupla en la tabla *Tabla*
            - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
            - Verifica si el valor de *Atributo* es único en la tabla *Tabla*, contando (count) la cantidad de tuplas que contienen dicho valor.
            - *(Si la cantidad de tuplas con el valor de Atributo es mayor a uno, el*

resultado es 0, de lo contrario 1)

- Inserta una tupla conteniendo *id_Error, id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*

- Sino, Si la *Funcion* es **CHECK_RULE**
  - Si el tipo de cálculo (dado por el parámetro *Param3*) es utilizando 'SQL'
    - Setea la sentencia SQL a aplicar, dada por el parámetro *Param2*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Verifica si la tupla cumple la sentencia SQL dada
      - *(Si cumple la sentencia, el resultado es 1, de lo contrario 0)*
      - Inserta una tupla conteniendo *id_Error, id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, Si el tipo de cálculo (dado por el parámetro *Param3*) es utilizando 'CATALOGO'
    - Setea el Catálogo a partir del parámetro *Param2*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Verifica si el valor de *Atributo* existe en el Catálogo
        - *(Si el valor de Atributo existe en el catálogo, el resultado es 1, de lo contrario 0)*
      - Inserta una tupla conteniendo *id_Error, id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, Si el tipo de cálculo (dado por el parámetro *Param3*) es utilizando 'PROCEDURE'
    - Setea el procedimiento a ejecutar a partir del parámetro *Param2*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Verifica - ejecutando el procedimiento – si la tupla cumple o no la regla esperada.
      - *(Si el procedimiento retorna 1, el resultado es 1, de lo contrario 0)*
      - Inserta una tupla conteniendo *id_Error, id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, retorna por excepción.
- Sino, Si la *Funcion* es **CHECK_CONSTRAINT**
  - Si el tipo de cálculo (dado por el parámetro *Param4*) es utilizando 'SQL'
    - Setea la sentencia SQL correspondiente a la condicion IF a partir del parámetro *Param2*
    - Setea la sentencia SQL correspondiente a la condicion THEN a partir del parámetro *Param3*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Verifica si la tupla no cumple la condicion IF o cumple la condicin IF y cumple la condicion THEN.
      - *(Si la tupla verifica la condición anterior, el resultado es 1, de lo contrario 0)*
      - Inserta una tupla conteniendo *id_Error, id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, Si el tipo de cálculo (dado por el parámetro *Param4*) es utilizando 'PROCEDURE'

- Setea el procedimiento a ejecutar a partir del parámetro *Param2*
- Para cada tupla en la tabla *Tabla*
  - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
  - Verifica - ejecutando el procedimiento – si la tupla cumple o no la condición esperada.
  - *(Si el procedimiento retorna 1, el resultado es 1, de lo contrario 0)*
  - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*

- Sino, retorna por excepción.

- Sino, Si la *Funcion* es **CHECK_LEVEL**
  - Setea la Tabla Nivel Precision (es decir la que asocia un valor de precisión a partir de un nivel de jerarquía) a partir del parámetro *Param3*
  - Si el tipo de cálculo (dado por el parámetro *Param4*) es utilizando 'TABLA'
    - Setea la tabla Referencial de Nivel a utilizar a partir del parámetro *Param2*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Obtiene - de la tabla Referencial de Nivel - el Nivel de jerarquía correspondiente al valor de *Atributo.*
      - A partir del Nivel de jerarquía anterior, obtiene - de la tabla Nivel Precision - el valor de precision a asignar.
      - *(en caso de corresponder más de un valor de precision para un mismo valor de Atributo, retorna el maximo de dichos valores de precisión; y en caso de no existir la correspondencia, retorna precisión 0)*
      - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, Si el tipo de cálculo (dado por el parámetro *Param4*) es utilizando 'SQL'
    - Setea la sentencia SQL a aplicar para obtener el nivel de jerarquia del valor, dada por el parámetro *Param2*
    - Para cada tupla en la tabla *Tabla*
      - Calcula el identificador de la tupla (*id_Tupla*) realizando un CHECKSUM de la tupla
      - Calcula el Nivel de jerarquía correspondiente al valor de *Atributo*, aplicando la sentencia SQL obtenida .*(Ejs: LEN(Atributo), (select case when Atributo is NULL then 1 when….else 3 from Tabla))*
      - A partir del Nivel de jerarquía anterior, obtiene - de la tabla Nivel Precision - el valor de precision a asignar.
      - *(en caso de corresponder más de un valor de precision para un mismo valor de Atributo, retorna el maximo de dichos valores de precisión; y en caso de no existir la correspondencia, retorna precisión 0)*
      - Inserta una tupla conteniendo *id_Error*, *id_Tupla* y el resultado de la verificación anterior en la tabla *Medida_Error*
  - Sino, retorna por excepción.

Anexo II – Detalle de las Dificultades encontradas y definiciones realizadas

Granularidad de la medición de exactitud

Se definió realizar la medición de exactitud a nivel de celda y tupla según corresponda en cada caso particular, y - con excepción del factor precisión - las métricas serán valores booleanos (0 o 1). Por simplificación del trabajo, se descartaron valores intermedios que en muchos casos podrían surgir de aproximaciones a un valor válido, por ejemplo en el caso de aproximación sintáctica a un valor válido.

Heterogeneidad de identificadores

La metadata a definir debía permitir almacenar el valor medido de exactitud asociado a cada tupla de cada tabla seleccionada del sistema fuente, donde cada tabla tiene su propia clave.

Para evitar la problemática relacionada a la heterogeneidad de los identificadores de dichas tablas, se optó por un identificador de tupla independiente de la definición de cada tabla. La implementación concreta de dicho identificador dependerá de las facilidades brindadas por el DBMS utilizado, por ejemplo en el caso de Oracle es conveniente la utilización del rowid, pues permite el acceso directo a la tupla, en cambio en MSSQLServer , por no existir un rowid predefinido, se utilizó la función CHECKSUM aplicada sobre la tupla para obtener el equivalente del rowid.

Utilización de funciones en el DBMS utilizado

Una de las definiciones secundarias realizadas fue la de independizar el procedimiento de medición de exactitud en las fuentes, de las funciones específicas a aplicar en cada caso, haciendo dicho procedimiento paramétrico e implementando cada función externamente al mismo. Lo anterior permitiría la incorporación de una nueva función agregándola en la metadata y creando la función a aplicar, sin alterar lo implementado.

Realizando un cambio en la configuración del DBMS se logró incorporar una función, pero debido a restricciones del DBMS, no fue posible utilizarla dentro del código SQL de forma de ser ejecutada para cada tupla seleccionada. La alternativa de implementar un procedimiento a ser ejecutado para cada tupla recorrida con un cursor, no obtenía una performance razonable.

Debido a que esta definición no era fundamental para los objetivos del trabajo y el DBMS podría no ser el definitivo, para no afectar los tiempos del trabajo se abandonó temporalmente la idea de independizar las funciones del procedimiento de medición. Por lo cual, se definió implementar las funciones dentro del procedimiento de medición, buscando resolverlas dentro de una consulta SQL para mejorar sustancialmente su performance.

Definición de Referenciales y Catálogos

Dificultades encontradas:

Representación del identificador de la instancia y su asociación con la tupla.

Para mantener una representación común a todos los Referenciales evitando la heterogeneidad de representaciones e identificadores, se optó por un identificador de tupla independiente de la definición de cada tabla. El identificador se obtiene mediante la aplicación de la función CHECKSUM sobre el o los campos que permiten identificar a la tupla y la instancia en el mundo real.

En el caso de los catálogos, no hay un identificador de instancia, pues es únicamente un diccionario de valores sintácticamente válidos, por lo tanto el identificador es el propio valor.

Representación de los sinónimos

Se discutió si la representación de los sinónimos debía ser externa a los Referenciales/Catálogos, finalmente se concluyó que ya estaba incluido en el primero y no era necesario en el segundo. Por lo tanto, en el caso de los Referenciales, los sinónimos simplemente son valores que están asociados a una misma instancia; y los catálogos contienen los sinónimos como otro valor válido.

En algún caso se creo una tabla de sinónimos temporal, a partir de la cual luego se ingresaron en el Referencial correspondiente. (Ej: Tabla Sinonimos_Lugares_Geograf: parejas de sinónimos, se utiliza para generar el referencial Referencial_Estudiantes para evaluar la correctitud semántica del Lugar de origen)

Definición de una estructura conveniente para estas tablas.

El campo a comparar se define con un nombre estándar ("Valor") independiente del campo destino y con igual tipo de datos que el campo destino contra el cual se realiza la comparación, para simplificar la implementación y el modelo. En el caso de los Referenciales, el identificador es tratado según lo definido en el punto 1, generándose también con un nombre estándar ("Id_Ref")

Heterogeneidad de DBMS y collations

Al involucrar distintos sistemas fuente y la base de datos utilizada para mantener la metadata, se pueden encontrar heterogeneidades a distintos niveles. Puesto que el objetivo del presente trabajo no se centra en la resolución de estas heterogeneidades, se busco su simplificación de forma de evitar este tipo de problemas.

Para evitar problemas relativos a la interconexión de DBMS y consultas distribuidas, se definió como DBMS a utilizar para el mantenimiento de la metadata, el mismo que el ya usado en los sistemas fuentes.

A su vez, la definición del conjunto de caracteres utilizados – dado por la definición del collation - y su ordenamiento puede no ser homogéneo entre los distintos sistemas. Debido a la necesidad de recuperar la base de datos fuente desde un respaldo previo, el collation de los datos recuperados difería del definido para la base de datos que almacena la metadata, lo cual introdujo errores de discrepancia de collations al cruzar tablas de las distintas bases de datos. Para evitar la carga de trabajo en una tarea periférica al proyecto, se optó por la solución más práctica, modificar únicamente las tablas puntuales que generan errores.

Definición de Check Constraint y Check Rule

Check Constraint

Para la implementación de la función CHECK_CONSTRAINT que mide un aspecto de la exactitud asociada al factor Inconsistencia, es necesario un mecanismo de especificación de restricciones. Para evitar tanto la definición de un lenguaje particular como su parseo y evaluación, se define la utilización del lenguaje SQL para tal fin.

Se optó por definir un sentido de la restricción y definirla mediante una estructura condicional IF/THEN, dada por una pareja de condiciones (**condicionIF, condicionTHEN**) donde para ser falsa la restricción, debe cumplirse la condicionIF y no cumplirse la condicionTHEN, de lo contrario la restricción es verdadera.

La exactitud determinada por la restricción estaría asociada a los atributos definidos únicamente en la "condicionIF" y esto esta dado por el sentido establecido. Si la condicionIF involucra un único atributo, entonces la restricción es a **nivel de atributo**, si involucra más de un atributo de la misma tabla se considera a **nivel de tupla**. En caso de incluir mas de un atributo de distintas tablas o funciones de agregación sobre atributos en la condicionIF, la restricción se considera a **nivel de base de datos** y no esta considerado el caso en este proyecto.

Check Rule

Para la implementación de la función CHECK_RULE , también es necesario un mecanismo de especificación de restricciones (en este caso reglas) a nivel de tupla . Nuevamente, para evitar tanto la definición de un lenguaje particular como su parseo y evaluación, se define la utilización del lenguaje SQL para tal fin.

A diferencia del caso anterior, se optó por definir una regla a evaluar como una única condición, sin necesidad de utilizar precondiciones.

Consideraciones sobre Foreigns Keys y Constraints

Al estudiar la función CHECK_FK que mide si son cumplidas las Foreign Keys asociada a un atributo de una tabla surgió la posibilidad de que dicha Foreing Key estuviera conformada por múltiples atributos. Considerar dicha situación, llevó a cuestionarse si la medición de exactitud en ese caso sería a nivel de grupo de celdas, de cada una de las celdas componente, o de tupla.

De forma similar en el caso de la función CHECK_CONSTRAINT para los atributos que participan en una restricción (constraint), ¿La medición realizada es a nivel de grupo de celdas, de cada una de las celdas componente o de tupla?

Se llegó a la conclusión que pese a que lo adecuado sería considerarlo a nivel de grupo de celdas, dado que en este trabajo no se contempla dicho nivel, se considerará a nivel de tupla.

Consideraciones sobre los niveles de precisión

Se discutieron distintas alternativas de medición de exactitud para el factor precisión. Entre ellas se consideró en particular dos casos:

- Que la precisión debía seguir algún tipo de jerarquía u orden parcial de niveles en la cual a mayor nivel, mayor precisión
- Que la precisión debía seguir algún tipo de jerarquía u orden parcial de niveles, pero el nivel óptimo es aquel que se corresponde con un nivel preseleccionado según el caso, y la lejanía o cercanía a dicho nivel establecía el valor de precisión. Este nivel preseleccionado como óptimo podría ser seleccionado tanto por el usuario final o por algún administrador.

Puesto que el primero puede considerarse un caso particular del segundo, se buscó un diseño que permitiera representar el segundo caso. Esto se logró mediante la especificación de la asociación entre cada nivel y su valor de precisión, en la tabla **Tabla_Medida_Precisión** de la metadata.

En el caso concreto de la tabla BD_LUGARES y el código del lugar, se implementó en base a la siguiente definición:

Nivel de precisión	Exactitud
5	0.2
4	0.25
3	0.33
2	0.5
1	1

La exactitud se calculó como 1/Nivel_Actual. De esta forma cuanto menor el nivel (mas cercano a uno), mayor la jerarquía de nivel y por lo tanto mayor es la exactitud. Además el ingreso de más niveles no invalida el cálculo de los anteriores, pues la formula es independiente de la cantidad de niveles

Identificadores de tupla duplicados

Al utilizar la función de CHECKSUM sobre cada tupla para generar el identificador de tupla, se encontró que en tablas con cierta cantidad de tuplas (más de 600.000) se generaban identificadores duplicados para distintas tuplas (cerca de 1.500 identificadores repetidos). Pese a conocer la posibilidad de ocurrencia de este problema, no se esperaba encontrarlo en esta magnitud, en la cantidad de tuplas manejada. El problema se vio acrecentado por la existencia de datos sucios (campos con valores nulos) lo cual incrementa las posibilidades de CHECKSUMs duplicados.

Se estudiaron alternativas, entre ellas:

1. Cambiar el DBMS por Oracle Database pues soporta nativamente el rowid, lo cual además mejoraría la performance al accederse directamente a la tupla (Sin acceso a través de índice o necesidad de recorrer toda la tabla) y se mantendría estable independientemente de los cambios realizados a cada tupla.
2. Agregar a las tablas una columna timestamp que simule el rowid; esto no se había realizado oportunamente para evitar modificar las tablas origen, manteniendo su autonomía, y para mantener un diseño simple.
3. Pasar a la función BINARY_CHECKSUM o una combinación de ambas, o incluso en algún caso crear una vista sobre la tabla origen agregando una columna con datos que generen mayor aleatoriedad al CHECKSUM generado, y de esta forma mejorar los resultados.
4. Dado que no es un problema central en nuestro trabajo, asumir que los CHECKSUM repetidos implica tuplas repetidas y tratarlo como un caso mas de inconsistencia 'Duplicated rows', ignorándola en nuestro caso o extendiendo el trabajo con una función CHECK_DUP que lo contemple.
5. Mantener un número secuencial global. Esta opción es similar a la opción 2, pero exige implementar un mecanismo de generación y almacenamiento de estos números (por ejemplo mediante triggers), con la ventaja que son estáticos y no se ven afectados por cualquier cambio en los datos de la tupla. Una forma de implementarlo podría ser utilizando unique identifiers (GUID), lo cual tiene la contra adicional de la lentitud en el acceso por ser uno de los tipos de datos más grandes del DBMS utilizado. Para evitar la implementación de dicho mecanismo, podría utilizarse un campo tipo identity, si no se utiliza uno ya en la tabla, por lo cual no es una solución homogénea.

La opción 1 parecía ser la más conveniente, pero también costosa en tiempo, pues ya se había avanzado en el trabajo y se partía a su vez de un trabajo previo y fuentes de datos en otro DBMS. La opción 2 es más práctica, pero afecta la autonomía de las fuentes y se ve afectada por cualquier cambio de datos en la tupla fuente, lo cual obliga a rehacer la medición de exactitud. La alternativa 3 es un workaround práctico al problema, aunque no soluciona el problema en sí , esto no es el objetivo central de este trabajo. Esta opción también se ve afectada por los cambios en la tupla del sistema fuente. La alternativa 4 es una simplificación al problema, pero mantiene el error en la medición de la exactitud, consecuencia del problema en la generación del identificador.

Por último, la alternativa 5 adolece de algunas desventajas de 2 y además de la necesidad de implementar un mecanismo para mantener los identificadores.

Con la meta de evitar afectar la autonomía de las fuentes y mantener un diseño simple y a su vez encontrar una solución rápida y simple, se optó por la opción 3, ignorar el error, dado que luego de efectuar la comparación a las posibles combinaciones, se encontró que utilizando la función CHECKSUM y BINARY_CHECKSUM y ubicando el campo generado al principio de la tupla para obtener mayor aleatoriedad, se obtienen 100 tuplas duplicadas de 600000, o sea un 0.016% de error, lo cual resulta aceptable. A su vez, de dichas tuplas, únicamente una se midió erróneamente su exactitud, es decir los otros 99 con identificador duplicado referencian a una tupla que tiene el mismo valor de exactitud, por lo cual a efectos de la medición el error fue 1/6000 %

Invalidez del identificador de tupla al realizar la actividad de ensuciar datos

En algunos casos particulares fue necesario ensuciar datos para lograr un valor de exactitud conveniente para la posterior propagación. Debido al procedimiento seguido, al modificar los datos asociados a un tipo de error, se

afectaba el CHECKSUM de las tuplas involucrados y por lo tanto del identificador de tupla. Esto invalidaba la medición realizada de los tipos de error previos, obligando a la rehacer la medición.

Para evitar este problema se planificó la actividad de medición y modificación de datos de forma de no afectarse mutuamente.

Función CHECK_CONSTRAINT : Precondición vs Condición

Según el criterio establecido, para efectuar la medición de exactitud se verifica primero la precondición de la Constraint (es decir, la condicionIF) y posteriormente se evalúa la condición. Si no se cumple la precondición se asigna el valor 1 de exactitud. Surge entonces el cuestionamiento respecto a si no cumplir la precondición invalida o no la Constraint, pero puesto que se sigue el criterio de un condicional IF-THEN, la clave esta en la adecuada definición de la precondición (condicionIF) y la condición (condicionTHEN).

Por ejemplo puede expresarse "if CampoX NOT NULL then CONDICION", con lo cual si CampoX es NULL, el valor de exactitud obtenido es 1, y otra manera podría ser "if TRUE then CampoX IS NOT NULL And CONDICION", en cuyo caso CampoX paso a formar parte de la condición, y si es NULL el valor de exactitud obtenido es 0. Ambas expresiones son válidas y tienen distinto significado, la primera no obliga a que el campoX sea no nulo para ser válida, simplemente indica que si es no nulo, entonces debe cumplirse la CONDICION; la segunda obliga a que el campoX sea nulo y que se cumpla la condición para ser válida.

Problema en la medición de exactitud.

Precondiciones adicionales para prevenir fallas de cálculo

En las funciones CHECK_RULE y CHECK_CONSTRAINT se definen condiciones a evaluar, pero puede ser necesario realizar chequeos previos para su validación.

Ej: CHECK_RULE

CC_CREDITOSMIN >= 0 AND CC_CREDITOSMIN <= 450

Falla si CC_CREDITOSMIN no es numérico o es nulo, por lo que se debe incorporar la siguiente precondición:

CC_CREDITOSMIN IS NOT NULL AND ISNUMERIC(CC_CREDITOSMIN)=1 AND

CC_CREDITOSMIN >= 0 AND CC_CREDITOSMIN <= 450

La precondición a chequear puede o no afectar el resultado de la condición a evaluar, o sea el cumplimiento de una precondición puede o no causar que el valor de exactitud para la condición sea 0, esto dependerá de cómo se plantee la condición. En el ejemplo anterior, podría expresarse lo siguiente, donde las precondiciones afectan de forma diferente al resultado final:

CC_CREDITOSMIN IS NULL OR ISNUMERIC(CC_CREDITOSMIN)<>1 OR (CC_CREDITOSMIN>= 0 AND CC_CREDITOSMIN <=450)

Ej: CHECK_CONSTRAINT

- precondición

Tabla.CampoX <> 'VALOR' AND (1)

Tabla.CampoY IS NOT NULL AND (2)

ISNUMBER(Tabla.CampoY) =1 (3)

- condición
Tabla.CampoY >= (Select min(CampoA) Tabla2)

(1) es la precondición y (2) y (3) son agregados necesarios para asegurar la capacidad de cálculo de la condición

Coherencia y completitud de los errores considerados

A partir de la observación de la necesidad de precondiciones adicionales para asegurar la posibilidad de cálculo de las condiciones a medir en las funciones CHECK_RULE y CHECK_CONSTRAINT, surge la reflexión sobre la necesidad de considerar los errores definidos no aisladamente, sino en conjunto, teniendo en cuenta la coherencia entre los mismos así como su completitud, de forma de lograr una medición de exactitud coherente y completa en su conjunto.

Es decir, los chequeos planteados no son independientes entre sí, por ejemplo en el caso considerado en el punto anterior, de no cumplirse la condición (2), es decir si Tabla.CampoY es nulo, no se cumple la precondición y por lo tanto se obtiene el valor 1 de exactitud. Sin embargo en este caso puede ser necesario tener en cuenta la medición realizada con la función CHECK_NULL para dicho campo, de lo contrario podrían obtenerse resultados incoherentes en su conjunto.

Por otro lado, si se elige un subconjunto de tablas y/o errores a considerar, podría suceder que su propagación no sea coherente por ser incompleto el conjunto considerado al ser necesario contar con un valor de exactitud para alguna tabla o error no considerado. Por ejemplo, si en la propagación se realiza un join de dos tablas y para una de ellas no se cuenta con información de exactitud o no se cuenta con la información para el tipo de error considerado.

Anexo III – Detalle de los resultados obtenidos

Para obtener los resultados presentados en la Tabla 2, se consideró que una tupla tiene exactitud 0 si existe un valor de exactitud 0 en alguno de sus atributos para alguna de las funciones aplicadas en la medición (de aquellas asociadas al factor de calidad considerado). La agregación de tupla a tabla se realizó promediando para cada tabla y factor de calidad considerado, los valores de exactitud de cada tupla obtenidos con el criterio antes mencionado.

Tabla	Consistencia	Sintáctico	Semántico	Precisión
AX_MAPEO_ASIGNATURAS	0.897268			
AX_MAPEO_CARRERAS	0.245283			
BD_ACTIVIDADES_VIEW	0.938552	0.874075		
BD_ASIGNATURAS	0.308788	0.9519	0.887173	
BD_CARRERAS	0.021276	0.553191	0.80851	
BD_EST_CARR	0.931623	0.943443		
BD_ESTUDIANTES	0.752688	0.540434	0.636556	
BD_INSTITUTOS		0.444444	0.722222	
BD_LUGARES		0.939393	0.90909	0.416666
BD_MAT_CARR	0.924302	0.482071		
BD_MATERIAS		0.521912	0.434262	

Tabla	Consistencia	Sintáctico	Semántico	Precisión
IN_CARRERAS			0.829787	

Tabla 2– Valores de exactitud agregados por Tabla y Factor de Calidad

Además se obtuvo información de la distribución de la exactitud medida, en función del factor de calidad y la función utilizada, para cada tabla / atributo (Tabla 4). Se agregó esta información a un mayor nivel de granularidad y se muestran los resultados en la Tabla 5 y 6.

Factor de calidad	Funcion Utilizada	Tabla	Atributo	Exactitud	% de contribución al Factor	% del Total de tuplas
CONSISTENCIA	CHECK_CONSTRAINT	AX_MAPEO_CARRERAS	CC_CODCARR	.000	.01	75.47
CONSISTENCIA	CHECK_CONSTRAINT	AX_MAPEO_CARRERAS	CC_CODCARR	1.000	.00	24.53
CONSISTENCIA	CHECK_CONSTRAINT	BD_ACTIVIDADES_VIEW	AC_TIPOACTIVIDAD	.000	5.44	6.14
CONSISTENCIA	CHECK_CONSTRAINT	BD_ACTIVIDADES_VIEW	AC_TIPOACTIVIDAD	1.000	83.10	93.86
CONSISTENCIA	CHECK_CONSTRAINT	BD_ESTUDIANTES	ES_GENERACION	.000	.24	8.56
CONSISTENCIA	CHECK_CONSTRAINT	BD_ESTUDIANTES	ES_GENERACION	1.000	2.60	91.44
CONSISTENCIA	CHECK_CONSTRAINT	BD_MAT_CARR	MA_CREDITOSMIN	.000	.00	7.57
CONSISTENCIA	CHECK_CONSTRAINT	BD_MAT_CARR	MA_CREDITOSMIN	1.000	.03	92.43
CONSISTENCIA	CHECK_FK	AX_MAPEO_ASIGNATURAS	CC_CODCARR	.000	.03	10.27
CONSISTENCIA	CHECK_FK	AX_MAPEO_ASIGNATURAS	CC_CODCARR	1.000	.22	89.73
CONSISTENCIA	CHECK_FK	BD_EST_CARR	CC_CODCARR	.000	.36	6.84
CONSISTENCIA	CHECK_FK	BD_EST_CARR	CC_CODCARR	1.000	4.86	93.16
CONSISTENCIA	CHECK_FK	BD_ESTUDIANTES	LU_CODLUGAR	.000	.51	17.83
CONSISTENCIA	CHECK_FK	BD_ESTUDIANTES	LU_CODLUGAR	1.000	2.34	82.17
CONSISTENCIA	CHECK_UNIQUE	BD_ASIGNATURAS	AS_CODAS	.000	.17	69.12
CONSISTENCIA	CHECK_UNIQUE	BD_ASIGNATURAS	AS_CODAS	1.000	.08	30.88
CONSISTENCIA	CHECK_UNIQUE	BD_CARRERAS	CC_CODCARR	.000	.01	97.87
CONSISTENCIA	CHECK_UNIQUE	BD_CARRERAS	CC_CODCARR	1.000	.00	2.13
PRECISION	CHECK_LEVEL	BD_LUGARES	LU_CODLUGAR	.170	3.03	3.03
PRECISION	CHECK_LEVEL	BD_LUGARES	LU_CODLUGAR	.200	3.03	3.03
PRECISION	CHECK_LEVEL	BD_LUGARES	LU_CODLUGAR	.250	3.03	3.03
PRECISION	CHECK_LEVEL	BD_LUGARES	LU_CODLUGAR	.330	33.33	33.33
PRECISION	CHECK_LEVEL	BD_LUGARES	LU_CODLUGAR	.500	57.58	57.58
SEMANTICO	CHECK_REF	BD_ASIGNATURAS	AS_NOMAS	.000	.89	11.28
SEMANTICO	CHECK_REF	BD_ASIGNATURAS	AS_NOMAS	1.000	6.97	88.72
SEMANTICO	CHECK_REF	BD_CARRERAS	CC_NOMCARR	.000	.04	19.15
SEMANTICO	CHECK_REF	BD_CARRERAS	CC_NOMCARR	1.000	.18	80.85
SEMANTICO	CHECK_REF	BD_ESTUDIANTES	LU_CODLUGAR	.000	32.82	36.34
SEMANTICO	CHECK_REF	BD_ESTUDIANTES	LU_CODLUGAR	1.000	57.47	63.66

Factor de calidad	Funcion Utilizada	Tabla	Atributo	Exactitud	% de contribución al Factor	% del Total de tuplas
SEMANTICO	CHECK_REF	BD_INSTITUTOS	IN_NOMINST	.000	.02	27.78
SEMANTICO	CHECK_REF	BD_INSTITUTOS	IN_NOMINST	1.000	.06	72.22
SEMANTICO	CHECK_REF	BD_LUGARES	LU_NOMLUGAR	.000	.01	9.09
SEMANTICO	CHECK_REF	BD_LUGARES	LU_NOMLUGAR	1.000	.14	90.91
SEMANTICO	CHECK_REF	BD_MATERIAS	MA_NOMMAT	.000	.66	56.57
SEMANTICO	CHECK_REF	BD_MATERIAS	MA_NOMMAT	1.000	.51	43.43
SEMANTICO	CHECK_REF	IN_CARRERAS	NOMCARR	.000	.04	17.02
SEMANTICO	CHECK_REF	IN_CARRERAS	NOMCARR	1.000	.18	82.98
SINTACTICO	CHECK_NULL	BD_ACTIVIDADES_VIEW	ES_CI	.000	1.14	2.49
SINTACTICO	CHECK_NULL	BD_ACTIVIDADES_VIEW	ES_CI	1.000	44.49	97.51
SINTACTICO	CHECK_NULL	BD_CARRERAS	CC_CREDITOSMIN	.000	.00	12.77
SINTACTICO	CHECK_NULL	BD_CARRERAS	CC_CREDITOSMIN	1.000	.00	87.23
SINTACTICO	CHECK_NULL	BD_ESTUDIANTES	ES_NROEST	.000	.20	13.45
SINTACTICO	CHECK_NULL	BD_ESTUDIANTES	ES_NROEST	1.000	1.27	86.55
SINTACTICO	CHECK_NULL	BD_MAT_CARR	MA_CREDITOSMIN	.000	.01	51.79
SINTACTICO	CHECK_NULL	BD_MAT_CARR	MA_CREDITOSMIN	1.000	.01	48.21
SINTACTICO	CHECK_RULE	BD_ACTIVIDADES_VIEW	AC_TIPORESULTADO	.000	4.61	10.10
SINTACTICO	CHECK_RULE	BD_ACTIVIDADES_VIEW	AC_TIPORESULTADO	1.000	41.02	89.90
SINTACTICO	CHECK_RULE	BD_ASIGNATURAS	AS_CREDITOSAS	.000	.01	4.81
SINTACTICO	CHECK_RULE	BD_ASIGNATURAS	AS_CREDITOSAS	1.000	.12	95.19
SINTACTICO	CHECK_RULE	BD_CARRERAS	CC_CREDITOSMIN	.000	.00	21.28
SINTACTICO	CHECK_RULE	BD_CARRERAS	CC_CREDITOSMIN	1.000	.00	78.72
SINTACTICO	CHECK_RULE	BD_CARRERAS	CC_NOMCARR	.000	.00	29.79
SINTACTICO	CHECK_RULE	BD_CARRERAS	CC_NOMCARR	1.000	.00	70.21
SINTACTICO	CHECK_RULE	BD_EST_CARR	EC_FECHAING	.000	.15	5.66
SINTACTICO	CHECK_RULE	BD_EST_CARR	EC_FECHAING	1.000	2.54	94.34
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_CI	.000	.37	25.05
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_CI	1.000	1.10	74.95
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_FECHANAC	.000	.07	4.62
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_FECHANAC	1.000	1.40	95.38
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_NROEST	.000	.34	23.12
SINTACTICO	CHECK_RULE	BD_ESTUDIANTES	ES_NROEST	1.000	1.13	76.88
SINTACTICO	CHECK_RULE	BD_INSTITUTOS	IN_NOMINST	.000	.00	55.56
SINTACTICO	CHECK_RULE	BD_INSTITUTOS	IN_NOMINST	1.000	.00	44.44
SINTACTICO	CHECK_RULE	BD_LUGARES	LU_NOMLUGAR	.000	.00	6.06
SINTACTICO	CHECK_RULE	BD_LUGARES	LU_NOMLUGAR	1.000	.00	93.94
SINTACTICO	CHECK_RULE	BD_MATERIAS	MA_NOMMAT	.000	.01	47.81
SINTACTICO	CHECK_RULE	BD_MATERIAS	MA_NOMMAT	1.000	.01	52.19

Tabla 4– Distribución de exactitud entre Factores/Función para cada tabla/atributo

Las Tablas 5 y 6 no son una simple agregación de la información de la Tabla 4, dado que de ser así se considerarían algunas tuplas más de una vez (por ejemplo, al aplicarse distintas funciones sobre las mismas tuplas). Se consideran entonces, la exactitud de todas las tuplas afectadas.

Factor de calidad	Funcion	Exactitud	% del Total de tuplas
CONSISTENCIA	CHECK_CONSTRAINT	0	6
CONSISTENCIA	CHECK_CONSTRAINT	1	94
CONSISTENCIA	CHECK_FK	1	89
CONSISTENCIA	CHECK_FK	0	11
CONSISTENCIA	CHECK_UNIQUE	1	30
CONSISTENCIA	CHECK_UNIQUE	0	70
PRECISION	CHECK_LEVEL	0,25	3
PRECISION	CHECK_LEVEL	0,5	58
PRECISION	CHECK_LEVEL	0,33	33
PRECISION	CHECK_LEVEL	0,17	3
PRECISION	CHECK_LEVEL	0,2	3
SEMANTICO	CHECK_REF	1	66
SEMANTICO	CHECK_REF	0	34
SINTACTICO	CHECK_NULL	0	3
SINTACTICO	CHECK_NULL	1	97
SINTACTICO	CHECK_RULE	1	89
SINTACTICO	CHECK_RULE	0	11

Tabla 5– Distribución de exactitud entre Factores/Función

Factor de calidad	Exactitud	% del Total de tuplas
CONSISTENCIA	0	7
CONSISTENCIA	1	93
PRECISION	0,17	3
PRECISION	0,2	3
PRECISION	0,25	3
PRECISION	0,33	33
PRECISION	0,5	58
SEMANTICO	0	34
SEMANTICO	1	66
SINTACTICO	0	7
SINTACTICO	1	93

Tabla 6– Distribución de exactitud entre Factores