

Manual Técnico

Tema: Servicio Base de Datos

Tabla de contenido

[Introducción 3](#_Toc473413908)

[Operaciones del servicio 6](#_Toc473413909)

[findById 6](#_Toc473413910)

[findByIds 7](#_Toc473413911)

[findByField 8](#_Toc473413912)

[findByQuery 9](#_Toc473413913)

[findByNativeQuery 10](#_Toc473413914)

[findAll 11](#_Toc473413915)

[getRelationObject 12](#_Toc473413916)

[create 13](#_Toc473413917)

[update 14](#_Toc473413918)

[delete 15](#_Toc473413919)

[Cliente JAVA 16](#_Toc473413920)

[findById 16](#_Toc473413921)

[findByIds 17](#_Toc473413922)

[findByField 18](#_Toc473413923)

[findByQuery 19](#_Toc473413924)

[findByNativeQuery 20](#_Toc473413925)

[findAll 21](#_Toc473413926)

[getRelationObject 22](#_Toc473413927)

[create 23](#_Toc473413928)

[update 24](#_Toc473413929)

[delete 25](#_Toc473413930)

Introducción

El servicio de base de datos es el encargado de permitir acceder a la base de datos desde los procesos desarrollados en bonita 7, el mismo ofrece la posibilidad de consultar, insertar, editar y eliminar registros de la misma utilizando Hibernate y JPA lo cual permite manipular la información de forma similar a como lo propone la BDM de bonita 7.

Este servicio pretende extender la BDM de bonita 7 permitiendo utilizar distintos Objetos de Negocio por proyecto e independizar su uso en los procesos que se requiera, de esta forma se logra que un proceso solo conozca de sus Objetos de Negocio y no los de otro.

El mismo ha sido desarrollado para dar soporte a la nueva plataforma de procesos de bonita 7 y se encuentra desplegado en el AS del Carbon.

Arquitectura

Este servicio está desarrollado bajo una arquitectura de 3 capas la cual está soportada por los frameworks Hibernate, Spring y Jackson para la conversiones a JSON, el objetivo es exponer recursos REST los cuales puedan ser consumidos desde JavaScript (formularios de bonita), Java (conectores de procesos) y aplicaciones nativas de celulares

.

Para consumir el servicio desde JavaScript se deben utilizar llamados de tipo XMLHttpRequest y para consumirlo desde Java se debe utilizar el cliente desarrollado por Trycore para tal fin el cual detallaremos más adelante.

Este servicio posee una particularidad con respecto a los Objetos de Negocio y es que los mismos están desacoplados del servicio, esto se diseñó de esta forma para permitir distribuirlos mediante una librería de Java, esto permite incluirlos como dependencia en el estudio de bonita 7 y utilizarlos dentro de los conectores a la hora de comunicarse con la base a través del cliente java mencionado.

A continuación el diagrama de arquitectura:



Operaciones del servicio

En esta sección describiremos todas las funcionalidades que posee el servicio así como la lista de parámetros y retorno que genera cada una, cabe destacar que este servicio está diseñado para trabajar con mensajes JSON por lo que el cuerpo de las solicitudes esperadas y de las respuestas que se retornen tendrán este tipo de contenido.

Las operaciones tienen diferentes tipos de parámetros de acuerdo el tipo de solicitud (GET; POST, PUT, DELETE), algunos parámetros son parte de la URL y están rodeados por llaves los cuales llamaremos PathVariable, luego están los que forman parte del QueryString, los que van luego del signo de interrogación, estos los llamaremos RequestParam para los llamados GET y finalmente están los parámetros en viajan en el body de la solicitud, también los llamaremos RequestParam pero son utilizados para los llamados POST y PUT.

## findById

Realiza la búsqueda de un tipo de Objeto de Negocio por su Id

**Método**: GET

**Ruta de acceso**: /businessData/{businessDataType}/{id}

**Parámetros**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| Id | Identificador del registro correspondiente al tipo de dato definido anterior | File | Si |

**Respuesta**: Objeto correspondiente al identificador, null si no lo encuentra.

## findByIds

Permite buscar varios objetos de negocio por varios identificadores

**Método**: GET

**Ruta de acceso**: /businessData/ {businessDataType}/findByIds?ids=…

**Parámetros**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| ids | Identificadores de los objetos a buscar separados por coma | String | Si |

**Respuesta**: Lista con los objetos correspondientes a los ids enviados, si no encuentra ninguno retorna una lista vacía.

## findByField

Realiza una búsqueda de objetos de negocio filtrando por el campo indicado y los valores enviados.

**Método:** GET

**Ruta de acceso:**  /businessData/{businessDataType}/findByField

/{fieldName}?f=…

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| fieldName | Nombre del campo sobre el cual realizar el filtro, el mismo debe ser un atributo de la clase enviada en el parámetro businessDataType |  |  |
| f | Valores sobre los cuales filtrar separados por comas | String | Si |

**Respuesta**: Lista con los objetos encontrados con el nombre de campo y los valores enviados, si no hay coincidencias retorna una lista vacía.

## findByQuery

Ejecuta una consulta JPQL customizada, estas consultas se pueden predefinir en la tabla WS\_CUSTOM\_QUERY del esquema BBVA\_BONITA, la sintaxis de la misma como se menciono debe estar expresada en formato JPQL utilizando alguno de los objetos de negocio de las bdm publicadas.

**Método:** GET

**Ruta de acceso:**  /businessData/{businessDataType}/findByQuery

/{queryName}?p=…&c=…&f=…

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| queryName | Nombre de la consulta a ejecutar, debe coincidir con el campo ‘name’ de la tabla WS\_CUSTOM\_QUERY | String | Si |
| p | Página a retornar, utilizado para paginación | Integer | Si |
| c | Cantidad de registros de la página | Integer | Si |
| f | Filtros de la consulta, si se desea expresar más de una repetir el filtro cuantas veces sea necesario | String | No |

*Para utilizar el filtro f la sintaxis es f=NOMBRE\_PARAMETRO\_QUERY=VALOR, para agregar otro filtro aplicar el carácter & e ingresar nuevamente la misma sintaxis.*

**Respuesta**: Lista con los objetos de negocio que encuentre la ejecución de la consulta, si no hay coincidencias retorna una lista vacía.

## findByNativeQuery

Operación similar a la anterior con la diferencia que la búsqueda la realiza utilizando una consulta nativa SQL, la misma se debe predefinir en la tabla WS\_CUSTOM\_QUERY de igual forma que en la operación anterior pero con sintaxis SQL

**Método:** GET

**Ruta de acceso:**  /businessData/{businessDataType}/findByNativeQuery

/{queryName}?p=…&c=…&f=…

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| queryName | Nombre de la consulta a ejecutar, debe coincidir con el campo ‘name’ de la tabla WS\_CUSTOM\_QUERY | String | Si |
| p | Página a retornar, utilizado para paginación | Integer | Si |
| c | Cantidad de registros de la página | Integer | Si |
| f | Filtros de la consulta, si se desea expresar más de una repetir el filtro cuantas veces sea necesario | String | No |

*Para utilizar el filtro f la sintaxis es f=NOMBRE\_PARAMETRO\_QUERY=VALOR, para agregar otro filtro aplicar el carácter & e ingresar nuevamente la misma sintaxis*

**Respuesta**: Lista con los objetos de negocio que encuentre la ejecución de la consulta, si no hay coincidencias retorna una lista vacía.

## findAll

Busca todos los objetos de negocio por tipo de dato

**Método:** GET

**Ruta de acceso:**  /businessData/{businessDataType}/findAll?p=..&c=…

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| p | Página a retornar, utilizado para paginación | Integer | Si |
| c | Cantidad de registros de la página | Integer | Si |

**Respuesta**: Lista con todos los objetos de negocio encontrados, si no existe ninguno retorna una lista vacía.

## getRelationObject

Dado que se están utilizando objetos de negocio mapeados a tablas con el ORM Hibernate las relaciones entre estos pueden definirse de tipo EAGER o de tipo LAZY, en el segundo caso las mismas no cargan la relación por defecto con el objetivo de mejorar la performance de las consultas pero puede haber casos en los que el desarrollador requiera de esta información relacionada para lo cual se creó esta operación.

**Método:** GET

**Ruta de acceso:**  /businessData/{businessDataType}/{id}

/{relationFieldName}

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| id | Identificador del objeto de negocio principal | Integer | Si |
| relationFieldName | Nombre del atributo del que se desea recuperar la información relacionada, debe existir en la clase del businessDataType enviado | String | Si |

**Respuesta**: Retorna la información del atributo indicado, el mismo puede ser un objeto plano o un array dependiendo el tipo de relación de la entidad.

## create

Operación que permite crear registros en la base de datos.

**Método:** POST

**Ruta de acceso:** /businessData/{businessDataType}

**Body:** JSON con el objeto a insertar debe coincidir con la estructura del businessDataType enviado el mismo debe enviarse sin el atributo id.

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |

**Respuesta**: Retorna el objeto insertado completamente incluyendo el identificador autogenerado.

## update

Operación que permite editar registros de la base de datos.

**Método:** PUT

**Ruta de acceso:** /businessData/{businessDataType}

**Body:** JSON con el objeto a editar debe coincidir con la estructura del businessDataType enviado, el mismo debe enviarse con el atributo id del objeto que se desea modificar.

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |

**Respuesta**: Retorna el objeto insertado completamente incluyendo el identificado.

## delete

Operación que permite eliminar registros de la base de datos.

**Método:** DELETE

**Ruta de acceso:** /businessData/{businessDataType}/{id}

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | String | Si |
| id | Identificador del objeto de negocio a eliminar | Integer | Si |

**Respuesta**: Retorna el objeto insertado completamente incluyendo el identificado

Cliente JAVA

Dado que los servicios son de tipo REST con mensajes JSON se creó un cliente JAVA para consumir los servicios a través HTTPClient.

Para poder utilizarlo se debe importar la librería JAR llamada **dbaccess-client.jar** en el studio de bonita o en el proyecto java que se desee, una vez en el classpath se podrá utilizar la clase llamada **DBAccessClient** la cual contiene las operaciones que permiten invocar el servicio. A continuación detallaremos las operaciones existentes y los parámetros que espera cada una.

## findById

Esta operación invoca la operación findById del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| id | Identificador del objeto | Long | Si |

**Respuesta:** Objeto encontrado, el tipo de dato retornado corresponde a la clase enviada en el parámetro businessDataType

## findByIds

Esta operación invoca la operación findByIds del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| ids | Identificadores de los objetos | List<Long> | Si |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al businessDataType enviado

## findByField

Esta operación invoca la operación findByField del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| fieldName | Nombre del atributo sobre el que buscar | String | Si |
| filters | Lista de valores sobre con los que realizar la búsqueda | List<String> | Si |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al businessDataType enviado

## findByQuery

Esta operación invoca la operación findByQuery del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| queryName | Nombte de la consulta | String | Si |
| page | Página a retornar | Integer | Si |
| count | Cantidad de registros de la pagina | Integer | Si |
| filters | Lista de filtros, los mismos deben ser expresados con la sintaxis NOMBRE\_FILTRO=VALOR | List<String> | No |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al businessDataType enviado

## findByNativeQuery

Esta operación invoca la operación findByNativeQuery del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| queryName | Nombte de la consulta | String | Si |
| page | Página a retornar | Integer | Si |
| count | Cantidad de registros de la pagina | Integer | Si |
| filters | Lista de filtros, los mismos deben ser expresados con la sintaxis NOMBRE\_FILTRO=VALOR | List<String> | No |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al businessDataType enviado

## findAll

Esta operación invoca la operación findAll del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| page | Página a retornar | Integer | Si |
| count | Cantidad de registros de la pagina | Integer | Si |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al businessDataType enviado

## getRelationObject

Esta operación invoca la operación getRelationObject del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| id | Identificador del objeto de negocio correspondiente al businessDataType | Long | Si |
| relationFieldName | Nombre del atributo de la relación existente en la clase enviada en el parámetro businessDataType | String | Si |
| relationDataType | Tipo de dato del atributo de la relación | Class<?> | Si |

**Respuesta:** Lista de objetos encontrados, los tipos de objetos corresponden al relationDataType enviado

## create

Esta operación invoca la operación create del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| objectEntity | Objeto con la información a persistir el tipo de dato debe coincidir con la clase del businessDataType enviado. | Long | Si |

**Respuesta:** Objeto de tipo businessDataType con toda la información insertada.

## update

Esta operación invoca la operación update del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| objectEntity | Objeto con la información a persistir el tipo de dato debe coincidir con la clase del businessDataType enviado. | Long | Si |

**Respuesta:** Objeto de tipo businessDataType con toda la información actualizada

## delete

Esta operación invoca la operación delete del servicio detallada anteriormente recibe los siguientes parámetros:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Tipo** | **Obligatorio** |
| serviceUri | URI del servicio, se debe indicar protocolo, dns, puerto y contexto (por defecto es dbaccess-svc) | String | Si |
| businessDataType | Nombre completo de la clase del Objeto de Negocio, el mismo debe contener el package y nombre de la clase | Class<?> | Si |
| Id | Identificador del objeto a eliminar | Long | Si |

**Respuesta:** Sin retorno