
CLOUD SHINE
GLOBAL
L.L.P
YOUR TRUSTED PARTNER

CLOUD SHINE GLOBAL L.L.P

Oracle Billing & Revenue Management

BRM Entities

Storable Class,

Storable Objects

BRM Database Storage Model

Storable classes & Data Model relationship

BRM Datatypes

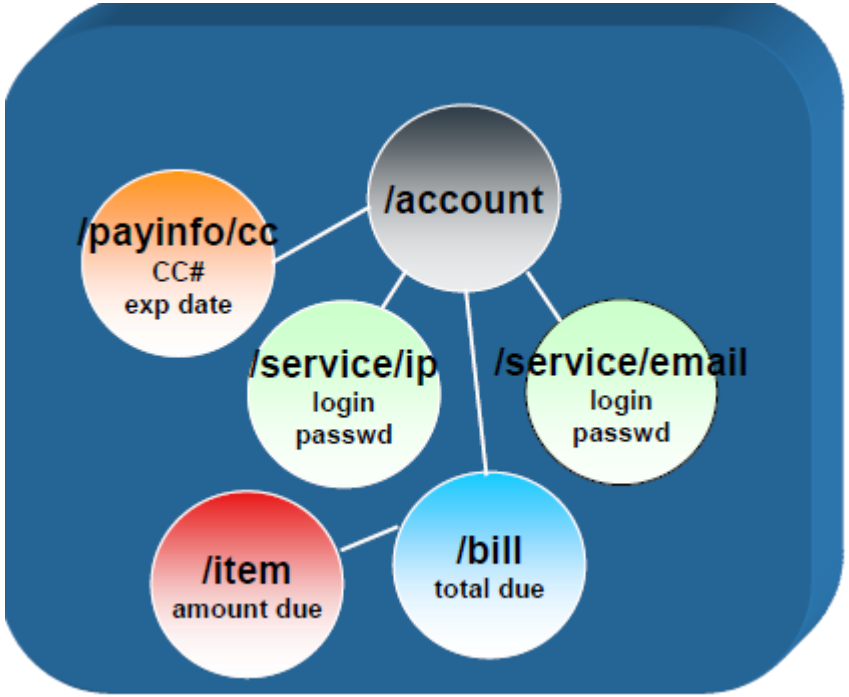
POID

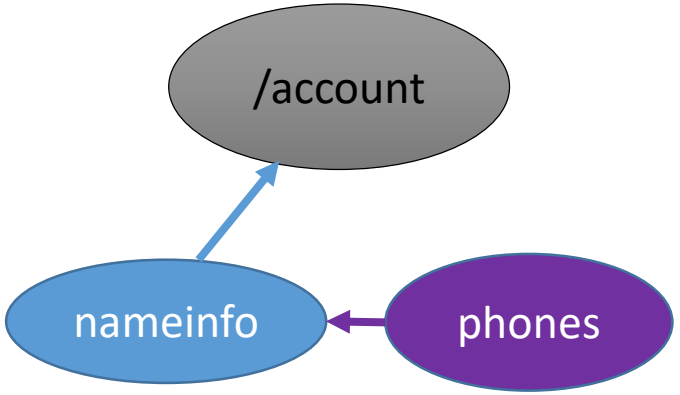
COMPLEX Data Structures

FLIST

BRM Entities

BRM application is designed on the real world telecom billing entities.
Entities helps to create different types of instances.
In BRM entities are **Storable classes** and instances of entities are called **Objects**.





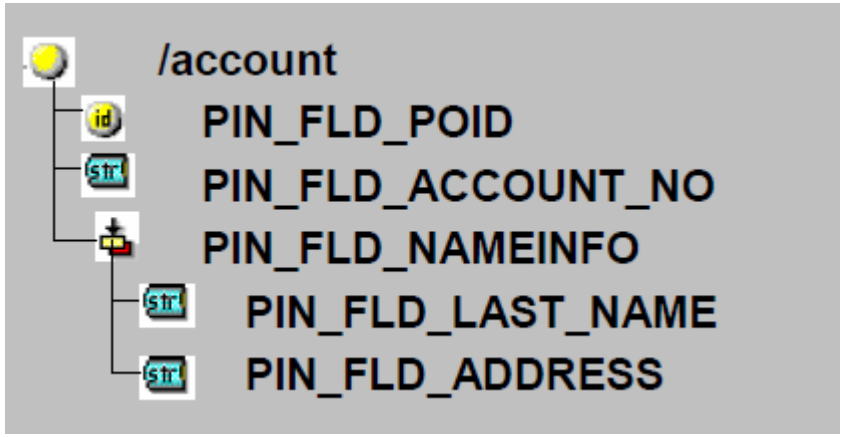
/account is the parent class. nameinfo and phones are subclasses.
Subclasses can be extended or non extended.
In BRM Storable classes are represented with forward slash (/) as prefix to storable class name.

Storable class

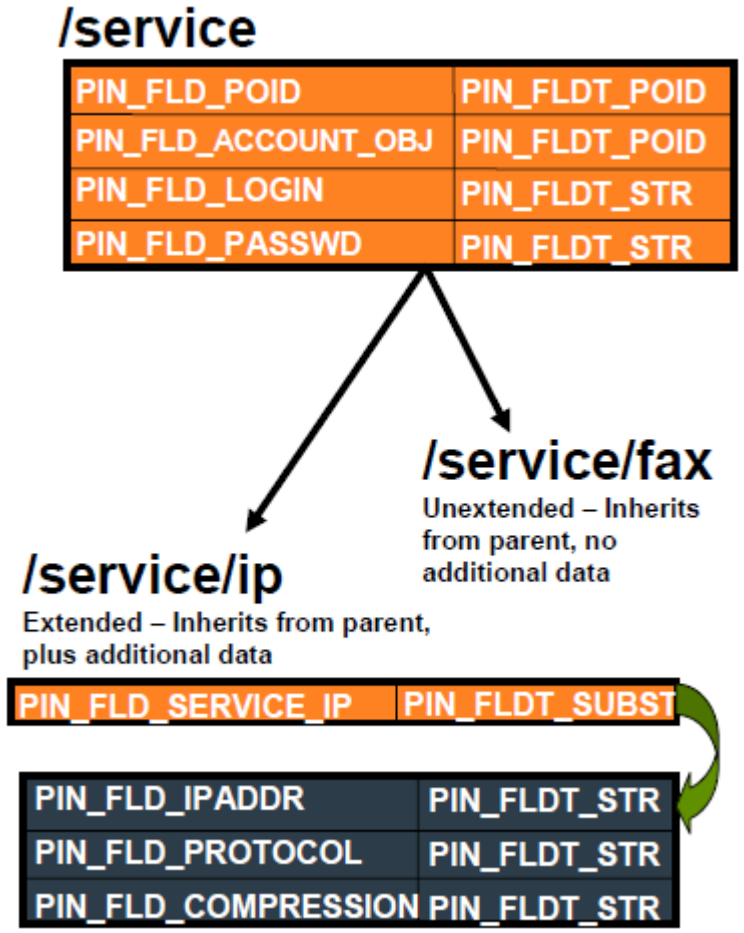
A **Storable Class** defines the structure for a set of data to be stored in the BRM Database.
It is just like a java class, as the structure is created in database, it is named as storable class.
Storable class fields can span more than one table.
Examples of some of important storable classes.

Storable class	Description	Fields span in tables
/account		
/balance_group		
/billinfo		

All storable class names start with "/".
Field names of storable classes are prefixed with PIN_FLD_XXXX. XXX stand for field name.



Storable class naming convention



All storable class names start with forward slash "/".

– Example: /service

Field names of storable classes are prefixed with PIN_FLD_XXXX. XXX stand for field name.

Subclasses are separated by another slash (/)

– Example: /service/ip, /service/fax

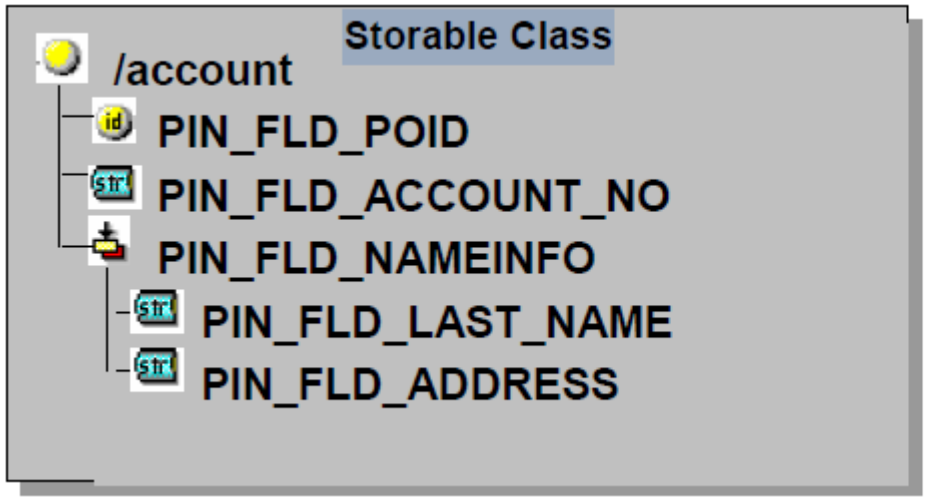
Subclass inherits parent class definition

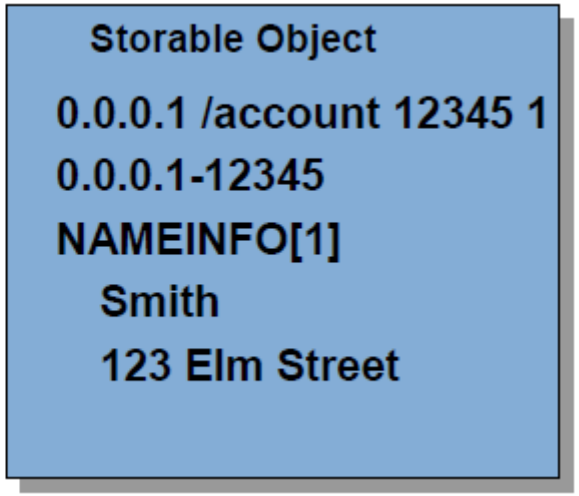
– Unextended subclasses have no additional data; same class definition as parent class.

– Extended subclasses have extended data in addition to data defined by parent class.

Storable Objects

A Storable object is an instance of the class, that is it represents actual set of data.
Example:
Student is a class and data with respective one student is an object.





CLOUD SHINE GLOBAL L.L.P

Important Storable classes

Storable Class	It's use
/account	Stores information about the customer, including contact names, address, status, and *customer segment* information. An **/account** object is linked to the following objects: Balance group objects that contain the account balances. Bill unit objects (**/billinfo**) that contain account billing information. Bill objects. A service object for each service that the account owns. An account can own any number of services. Additional account information stored in **/profile** objects.
/balance_group	Stores the balance information for various resources in an account such as dollars, free minutes, bytes, and frequent flyer miles. A balance group includes one or more sub-balances for each resource. The sub-balance contains the current amount, resource type, validity dates for the resource, rollover data, and sub-balance contributors.
/billinfo	Stores all billing, payment method, accounting cycle, payment collection date, and hierarchy information necessary to bill an account. A **/billinfo** object is created for every account.
/bill	Stores billing information, such as the amount due, amount adjusted, currency, and bill number.

CLOUD SHINE GLOBAL L.L.P

Important Storable classes Continued..

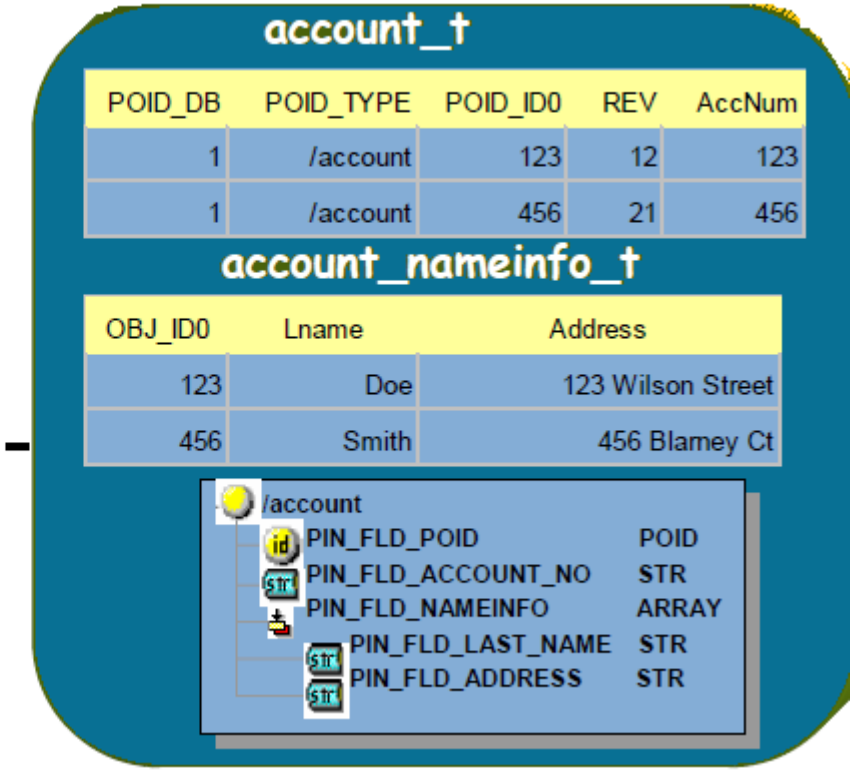
Storable Class	It's use
/item	Abstract class for storing *accounts receivable (A/R)* information. Subclasses of the **/item** object store different types of A/R information; for example, payments, adjustments, and cycle charges. Any impact to an account's A/R is stored in an **/item** object.
/invoice	Stores a customer invoice and information about the invoice, such as the bill it is associated with. Each **/bill** object can have a corresponding **/invoice** object.
/service	Abstract class to support subclasses for specific services, such as telephony or IP access. Subclasses define the properties that are specific to each service; for example, the telephony bearer service or the IP address.
/purchased_product	Stores information about purchased products for **/account** objects. Products owned by **/account** or any services in **/account** are stored in one or more of these objects. This object contains the reference to **/account**. **/account** has no references to this object.
/purchased_discount	Stores information about purchased discounts for **/account** objects. Discounts owned by an account or any services in the account are stored in one or more of these objects. This object stores the reference to the **/account** object; however, the **/account** object does not store a reference to this object.

BRM Database Storage Model

Data modeling (data modelling) is the process of creating a data model for the data to be stored in a database. This data model is a conceptual representation of Data objects, the associations between different data objects, and the rules. BRM database is no exceptional.
BRM Storage model consist of relational database and data dictionary

BRM Database
- Relational Database
- Tables and columns

Data Dictionary
- Storable Class Definitions
- Field Definitions



CLOUD SHINE GLOBAL L.L.P

Storable Classes & Data Model Relationship

Important Points About Storable classes

1. Storable classes can have no or more than one child/extended class.
2. All parent storable class have POID_id0 field which is the primary key to identify the object.
3. Subclasses are linked to parent/base classes in the table using obj_id0 field.
4. **obj_id0** field is mandatory in all subclasses. This field is linked with parent class **poid_id0** field.
5. Different storable classes are linked by having primary key of one storable class as foreign key in other parent classes.

   Example:

   /account is a storable class and **account_t** is the parent table for the storable class.

   poid_id0 of account_t is primary key.

   When an account is linked with other objects like **/bill** , table name **bill_t,** it will have **account_obj_id0** as foreign key.

6. **Sub tables will not have poid_id0.**
7. foreign key names will be **storableclassname_obj_id0.**

Storable Classes & Data Model Relationship

Storable Class	/account		
Table Name	account_t		

poid_id0	poid_type	rev	account_no
12345	/account	1	1100123
454556	/account	20	111001

Storable Class	/bill				
Table Name	bill_t				

poid_id0	poid_type	rev	bill_no	account_obj_id0	invoice_obj_id0
300100	/bill	1	B1-123	12345	0
43001	/bill	2	B2-321	454556	0

Subtable name	account_nameinfo_t		

obj_id0	first_name	last_name	street
12345	tom	cat	liverpool
454556	jerry	rat	liverpool

1. In the above example, account_t is linked to sub table through poid_id0 field of parent table[account_t] and obj_id0 [sub table]field.
2. In bill_t table, the account_t table is linked with foreign key field account_obj_id0.

CLOUD SHINE GLOBAL L.L.P

BRM Data Type

BRM has set of data types.
Simple data types generally correspond to C data types.

Data Type	Description
PIN_FLDT_INT	Signed Integer
PIN_FLDT_ENUM	Enumerated Integer
PIN_FLDT_DECIMAL	Decimal Number
PIN_FLDT_STR	Character String
PIN_FLDT_BINSTR	Binary String
PIN_FLDT_TSTAMP	Timestamp

Note: Fields should be defined in pcm_flds.h header file and also in data dictionary before they are used in flist.

CLOUD SHINE GLOBAL L.L.P

BRM Data Type

The complex data types are specific to BRM

- POID (Portal Object ID)
- Array
- Substruct
- Buffer

Data Type	Description
PIN_FLDT_POID	Portal Object ID
PIN_FLDT_ARRAY	Array (of nested Flists)
PIN_FLDT_SUBSTRUCT	Substructure (nested Flist)
PIN_FLDT_BUF	Arbitrary buffer of data

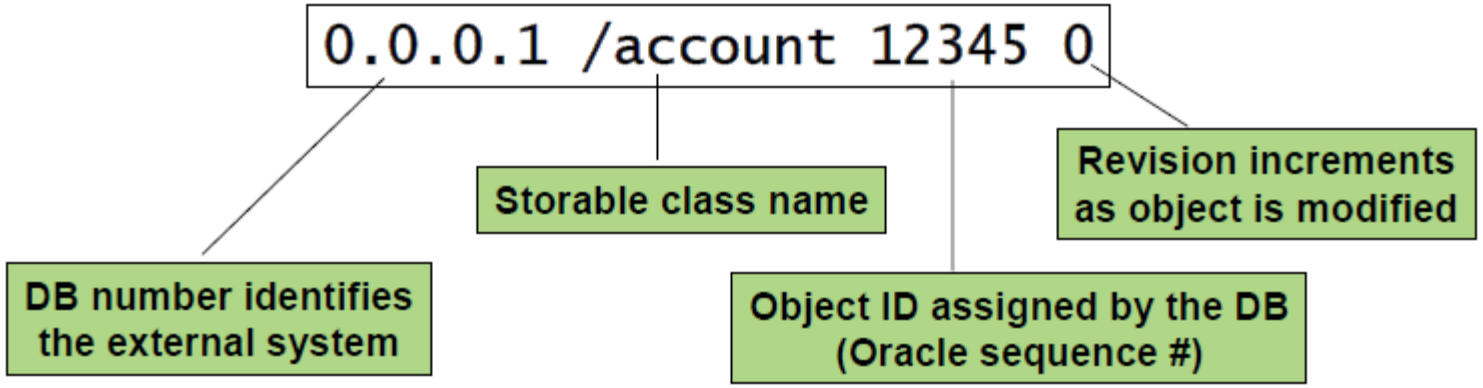
Note: Fields should be defined in pcm_flds.h header file and also in data dictionary before they are used in flist.

POID [Portal Object IDentifier]

POID – Represents a unique value. It is similar to primary key in database.
Uniquely identifies a set of data (storable object) in the BRM database.

A POID value contains 4 pieces of information

- Database number
- Object type (storable class name)
- Object ID
- Revision level



CLOUD SHINE GLOBAL L.L.P

Complex Data Type

SUBSTRUCT Data Type

A SUBSTRUCT is a reference to a nested Flist

Typically used with storable subclasses to define the extended set of data

– For performance reasons nested Flists should be limited to 3 levels

Used with Input and Output Flist specs to segregate sets of data

– No limit to nesting level

Array Data Type

An ARRAY References an array of nested Flists

Each Flist in the array is referenced by an Element ID

– Sparse arrays; element ID does not have to start with 0 or 1

– Element IDs do not have to be sequential

Each element in the array shares the same Flist specification

– Field name, value, nesting level, permissions

– An array with 1 element is functionally equivalent to a substruct (refers to a single nested Flist)

BRM Flist

BRM internal data structure is called Flist.
Flist stands for field value pair.
Used to pass data between functions also know as Opcodes.

Flist

Field	Value
Name	“TOM”
Age	45
CITY	“Delhi”
Height	172
Weight	75
County	“India”

BRM Flist

BRM application has developed using C, C++ and Java.
Simple data types generally correspond to C data types.
BRM business tier has a structure framework called Flist, which is used to process and pass data among different function modules internally.
Flist framework has list of fields and field values on it.
Flist framework has specifications which needs to be follow strictly to define the Flist.

Syntax of Flist Specifiation

Nesting Level	Field Name	Data Type	field value
0	PIN_FLD_POID	POID [0]	0.0.0.1 /account 12345

Nesting Level: Used to specify the field depth in the storable class hierarchy. All parent class fields will be at level 0 and immediate subclass or child class will be at the next level.
Field Name: Field name is the field's of storable class.
Data type: Any of the BRM data types
Field Value: Value of the field

CLOUD SHINE GLOBAL L.L.P

“Thank You”