1. **Bases de datos Objeto - Relacionales**

La gracia de estas bases de datos es poder crear tus propios objetos y métodos para ellos.

Además, se permite la gestión de tipos complejos de manera simple sin forzar los datos simples. Se pueden adaptar los modelos y las funciones e las bases de datos a este otro tipo.

Se proporciona mecanismos para que el usuario pueda definir sus propios tipos de datos cuya estructura puede ser compleja. Se reconoce el tipo de objeto, donde están localizados y tienen un identificador único.

1.1 **Datos definibles por el usuario.**

Se puede definir tipo de objeto o una colección.

**1.1.1 Objetos**

El objeto es un elemento del mundo real que tiene diferentes partes, un nombre que lo identifique y diferencie del resto, sus atributos que pueden modelar la estructura y los valores de ese tipo. Cada atributo puede ser de un tipo de datos básico o un tipo de datos definido por el usuario. Por último tendrá sus propios métodos, ya sea en MySQL o en C.

Para definir, los atributos hay que dar el tipo, el tamaño y las diferentes restricciones.

**1.1.1.1 Métodos**

Siempre se hace junto a la creción de su tipo, y debe llevar siempre una directiva de compilación.

WNDS: no se permite modificar las tablas

WNPS: se permite modificar las variables SQL PL

RNDS: no se permite leer las tablas

RNPS: se permite leer las variables SQL PL

Los métodos se ejecutarán sobre los objetos de su mismo tipo.

**1.1.1.1.1 Constructores**

Igual que en programación orientada a objetos.

**1.1.1.1.2 Métodos de comparación**

El método MAP sirve para indicar cual es de los atribtos del tipo para ordenar y comparar los objetos por medio de < y >. Permite decir que los objetos de tipo x se van a comparar por su atributo Y.

MAP MEMBER FUNCTION ret\_value RETURN NUMBER

PRAGMA ERSTRICT\_PREFERENCES