UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

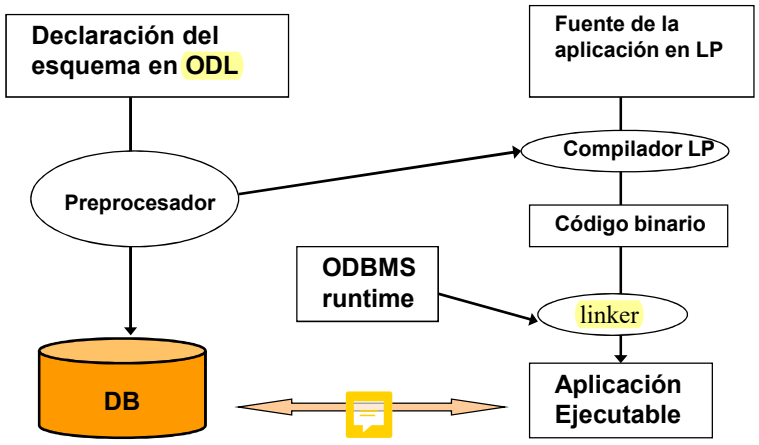
Aplicaciones y OQL

Dr. José Luis Zechinelli Martini
joseluis.zechinelli@udlap.mx
LIS – 3071

Este material está basado en el curso de la Profesora Christine Collet, Instituto Politécnico de Grenoble, Equipo NODS, LIG

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

ODMG: Aplicación



UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Object Query Language (OQL)

- OQL tiene una sintaxis a la SQL-2.
- OQL provee un acceso declarativo a los datos (OQL puede ser optimizado):
  - operaciones para seleccionar los objetos y valores.
  - operaciones para la actualización explicita usando las operaciones definidas sobre los objetos.
- OQL permite crear objetos (constructor) y valores.
- OQL puede ser utilizado de manera interactiva o dentro de un lenguaje anfitrión (puede llamar las operaciones definidas en el lenguaje).

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Esquema ejemplo (1)

```
struct Address {
    string street;
    string city;
    string code;
}
```

```
class Person    (extent Persons)
{   attribute string name;
    attribute enum gender {mal, female};
    attribute Address address;
    attribute set<string> phones;
    attribute Date birthdate;
    relationship Person spouse inverse Person::spouse;
    relationship Set<Person> children inverse Person::parents;
    relationship Set<Person> parents inverse Person::children;
    short  age();
    boolean lives_in(string);
    Person oldest_child();
    set<string> activities();
}
```

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Esquema ejemplo (2)

```
class Employee extends Person (extent Employees) {
        attribute Date hireDate;
        attribute float salary;
        attribute short deptno;

        relationship Set<Employee> subordinates inverse Employee::chef;
                …
        void seniority();
        void hire();
        void fire() raises (no_such_employee);

        set<string> activities();
}
```

Chairman un nombre haciendo referencia a un objeto de tipo `Person`.

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Ejemplos de consultas

Expresión OQL:	Tipo del resultado:
2	`short`
2 + 2	`short`
Chairman	`Person`
Persons	`set<Person>`
Chairman.name	`string`
Chairman.address.city	`string`
Chairman.phones	`set<string>`
Chairman.age	`short`
nil	`null`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Construcción de valores y objetos

- Una estructura: `struct( name: "Peter", age: 36 )`
- Un conjunto: `set( 3, 1, 2 )`
- Un conjunto múltiple: `bag( 3, 3, 1, 2, 2 )`
- Listas: `list( 1, 2, 3, 3 )`

- Construcción de objetos:

```
Person( name: "Pat", birthdate: Date '1956-3-8' )
Employee( name: "Michel", salary: 30000 )
```

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Select – from - where

- select p.age from Persons p
  where p.name = "Pat"
  - bag<short>

- select distinct p.age from Persons p
  where p. Name = "Pat"
  - set<short>

- select distinct struct(age: p.age,
  sex: p.gender)
  from Persons p
  where p.name = "Pat"
  - set<struct(age: short,
    sex: string)>

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Tipo resultante

- Un objeto:
  - Chairman
  - element(select p from Person p where p.name = “Pat”)
- Un valor:
  - Una literal
  - Una colección: `bag`, `set`, `list`

select distinct p.name from Persons p → set(string)

select [distinct] p.name from Persons p order by p.name → list(string)

select t from Persons t order by * → list(Person)

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Forma general select

select caminos / construcción de valores

from puntos de entrada (nombres de objetos o de extensiones, select)

[**where** predicado]

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Select: caminos

- // camino simple (atributo)

```
select  e. salary
from    Employees e
```

  - `bag<float>`

- // camino a través de una asociación

```
select  p.spouse.address.city
from    Persons p
```

  - `bag<string>`

- // camino en el **from**

```
select  distinct c. address
from    Persons p, p.Children c
```

  - `set<Address>`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Select: construcción de valores

```
select distinct struct(  name: e.name,
                         hps: ( select  y
                                from    e.subordinates as y
                                where   y.salary > 100000 ) )

from Employees e
```

- `set< struct( name: string, hps: bag<Employee> ) >`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

From: select

select struct(a: x.age, s: x.sex)

from (**select** e **from** Employees e **where** e.seniority = “10”) **as** x

where ...

- `bag< struct( a: short, s: gender ) >`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

From: *join*

Sea `Flowers: set<Person>`

select p

from Persons p, Flowers f

where f.name = p.name

- `bag<Person>`

Predicados

- **select** c.address
  from Persons p, p.children c
  where p.address.street = “René Char”
  and count(p.children) >= 2
  and c.address.city != p.address.city
  - `bag<struct Address>`

- **select** p.name
  from Persons p
  where p.address != nil
  and p.address.city = “Paris”
  - `bag<string>`

Valores indefinidos (1)

- **select** e
  from Employees e
  where e.address.city = “Paris”
  - `bag<Employee>`

- **select** e.address.city
  from Employees e
  - `bag<“Puebla”,“Cholula”, UNDEFINED>`

- **select** e.address.city
  from Employees e
  where is_defined(e.address.city)
  - `bag<string>`

Valores indefinidos (2)

- **select** e
  from Employees e
  where is_undefined(e.address.city)
  - `bag<Employee>` que no tienen dirección

- **select** e
  from Employees e
  where not(e.address.city = “Paris”)
  - `bag<Employee>` que no viven en Paris

Algunos operadores

- `count`, `min`, `max`, `sum`, `avg`. Ejemplos:
  `count(Persons), max(select salary from Employees)`
- `element`: para extraer el solo elemento de un conjunto.
- `listtoset`, `flatten`.
- `for all`: cuantificador universal:
  `for all e in Employees: e.salary > 100000`
- `exists`: cuantificador existencial.
- `like`: para comparar las cadenas.
- `define`: para nombrar el resultado de una consulta.
- Operadores sobre conjuntos: `union`, `intersect`, `except`.
- Pertenencia: `in`. Ejemplo: `Chairman in Persons`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Consultas usando métodos

- **select** max(**select** c.age
  from p.children c)
  from Persons p
  where p.name = “Charles“
  - `set<short>`
- **select** p.oldest_child.address.street
  from persons p
  where p.lives_in(“Grenoble“)
  - `bag<string>`
- Actividades definidas en `Person`, `Employee`, `Student`, …

  select p.activities **from** Persons p
  - `bag< set<string> >`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Especificidades SQL (Abreviaciones sintácticas)

- **select** p: name, salary, section_id
  from Professors p, p.teaches
  - `bag< struct( p: string, salary: float, section_id: string) >`
- **select** * **from** Persons p
  - `bag< struct( p: Person ) >`
- **select** count(*) **from** …

  Equivalente a: count(**select** * **from** …)

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Group by

- **select** *
  from Persons p
  group by p.age

  ➢ bag< struct(
  age: short,
  partition: bag< struct(p: Person) >)>

- **select** age,
  persons_num: count(partition)
  from Persons p
  group by p.age

  ➢ bag< struct(
  age: short,
  persons_num: unsigned short) >

- **select** city, partition:
  (select t.p from partition t)
  from Persons p
  group by p.address.city

  ➢ bag< struct(
  city: string,
  partition: bag<Person>) >

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Group by & having

- **select** city, partition: (**select** t.p
  from partition t)
  from Persons p
  group by p.address.city
  having count(partition) > 10000

  ➢ `bag<struct(`
  `city: string,`
  `partition: bag<Person>)>`

- **select** departement,
  avg: avg(**select** t.e.salary **from** partition t)
  from Employees e
  group by departement: e.deptno
  having avg(**select** t.e.salary
  from partition t) > 30000

  ➢ `bag<struct(`
  `departement: string,`
  `avg: real )>`

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Order by

Obtener una lista ordenada:

select p **from** Persons p **order by** p.name

select p **from** Employees p
order by p.age **desc**, p.name **asc**, p.salary

UDLAP
UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Define

Nombrar el resultado de una consulta:

- `define Q1 as element(select c from Persons c where c.name = "Yohan")` → `Person`
- `Q1` → `Person`
- `define Q2 as select e from Employees e where e.salary > 50000` → `bag<Employee>`
- `select t.name from t in Q2 order by *` → `list<string>`