

Property List

///BNF

<property_list ::== <property_spec>; | <property_spec><property_list> <property_spec>::= <attribute_spec> | <relationship_spec>

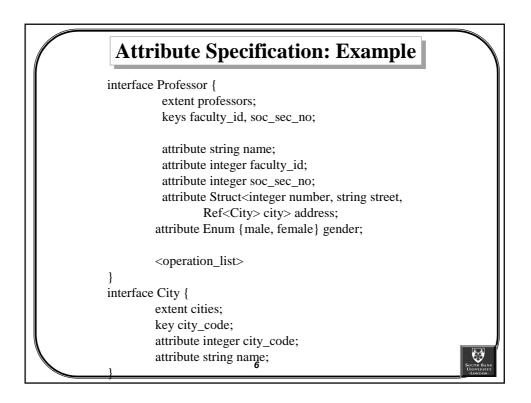
*#*Structured types have bracketed list of field-type pairs associated with them.

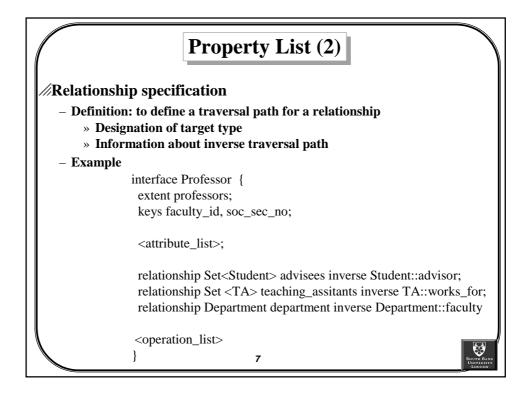
- //Enumerated types have bracketed lists of values.
- //Relationship have inverses.

//An element from another class is indicated by *<class>::*

//Form a set type with Set<type>.

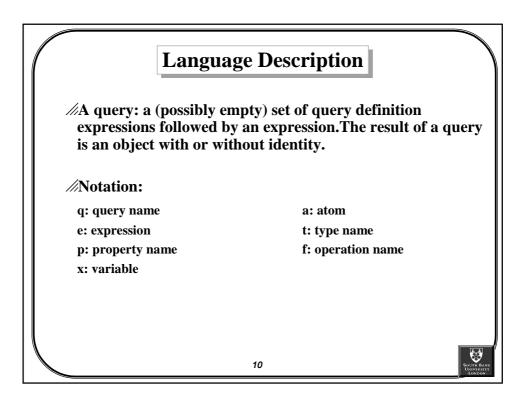
5

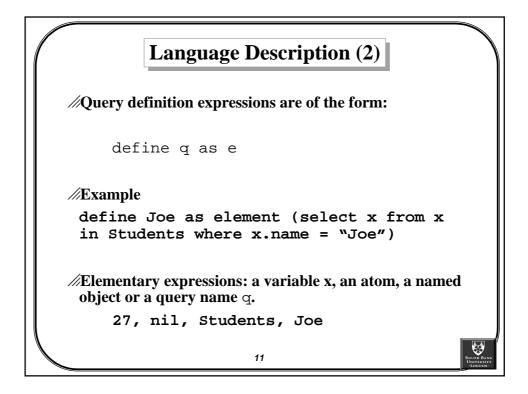


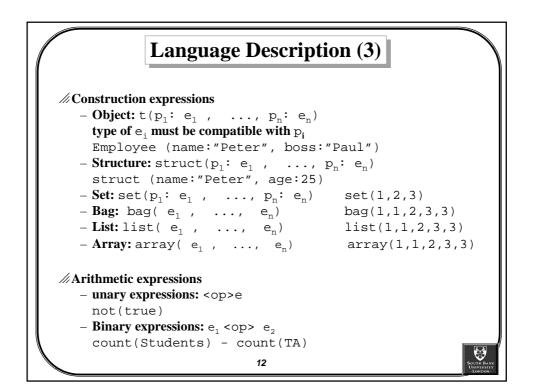


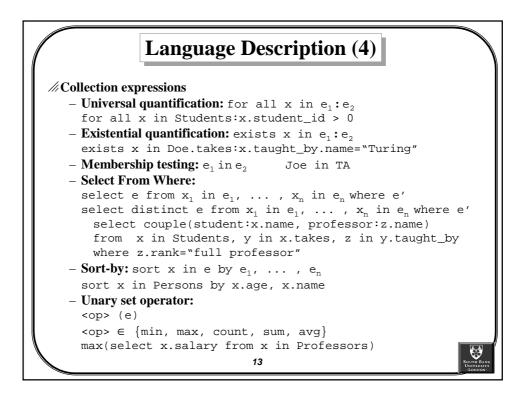


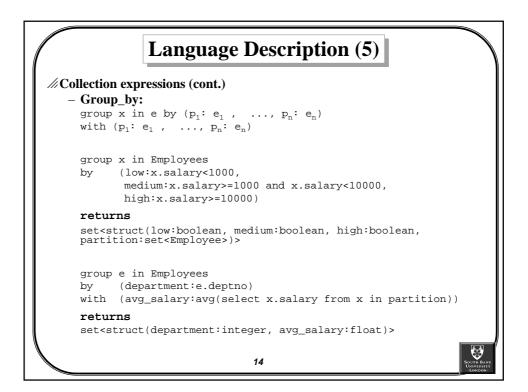
Object Query Language //Relies on ODMG object model //OQL is very close to SQL-92. Extensions concern object-oriented notions. //High level primitives to deal with sets of objects, structures and lists. **//OQL** is a functional language where operators can freely be composed, as long as the operands respect the type system. *//*Not computationally complete. //No explicit update operators (instead use operations defined on objects). //Declarative access. Thus OQL queries can be easily optimised by virtue of this declarative nature. //Formal semantics can easily be defined ŝ 9

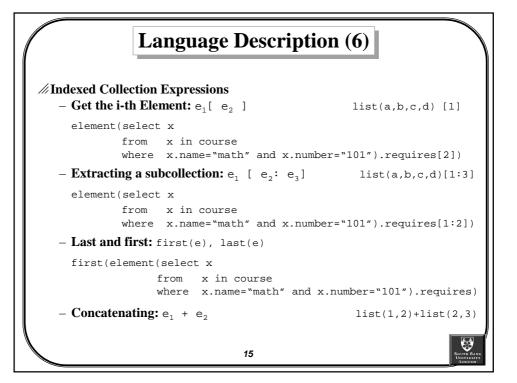


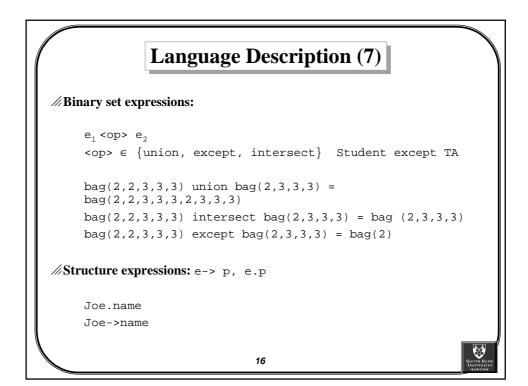


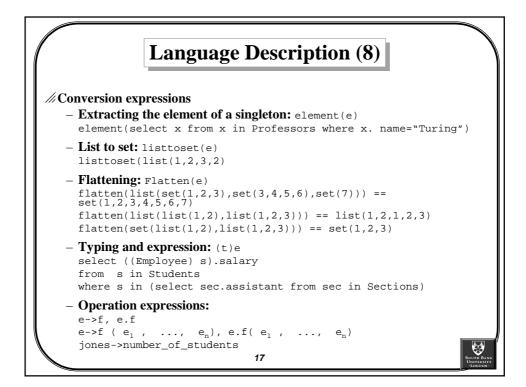


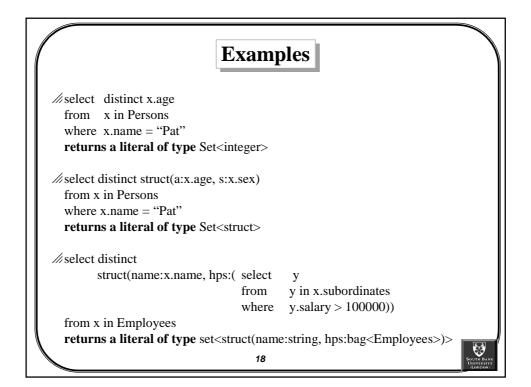












	Examples (2)
∬ select struct (a:x.age, from x in (select from	
where x.name = "Pat" returns a literal of ty	
<pre>// Chairman returns the Chairman object (just the one, presumably!)</pre>	
// Chairman.subordinates returns the set of subordinates of the Chairman	
// Persons returns the set of all persons	
	19 Lancom

