## Part 11: Object-Relational Constructs in Oracle

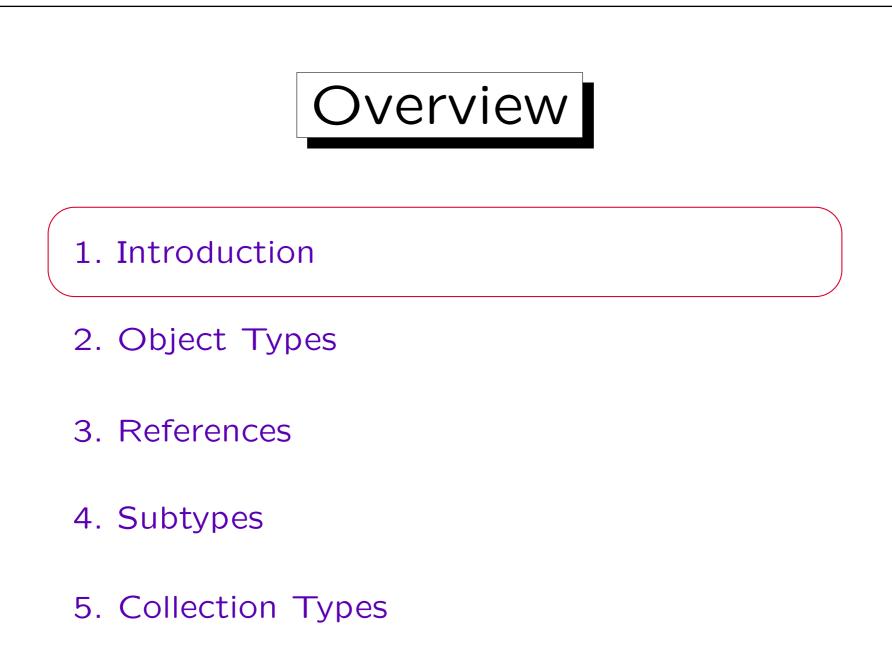
## **References:**

- Jeffrey D. Ullman: Object-Relational Features of Oracle [http://infolab.stanford.edu/ ullman/fcdb/oracle/or-objects.html]
- Jeffrey D. Ullman: CS145 Introduction to Databases (Autumn 2007) [http://infolab.stanford.edu/ ullman/fcdb/aut07/] Lecture Notes: Object-Relational SQL [http://infolab.stanford.edu/ ullman/fcdb/aut07/slides/or.pdf]
- Oracle: Database Object-Relational Developer's Guide [https://docs.oracle.com/database/121/ADOBJ/adobjint.htm#ADOBJ7025]
- Oracle: Oracle8 Concepts, 12. Using User-Defined Datatypes [https://docs.oracle.com/cd/A64702\_01/doc/server.805/a58227/ch\_objs.htm]
- Oracle: Advanced Topics for Oracle Objects [https://docs.oracle.com/cd/E11882\_01/appdev.112/e11822/adobjadv.htm]
- : Oracle: Using PL/SQL Object Types. [https://docs.oracle.com/cd/B13789\_01/appdev.101/b10807/10\_objs.htm]

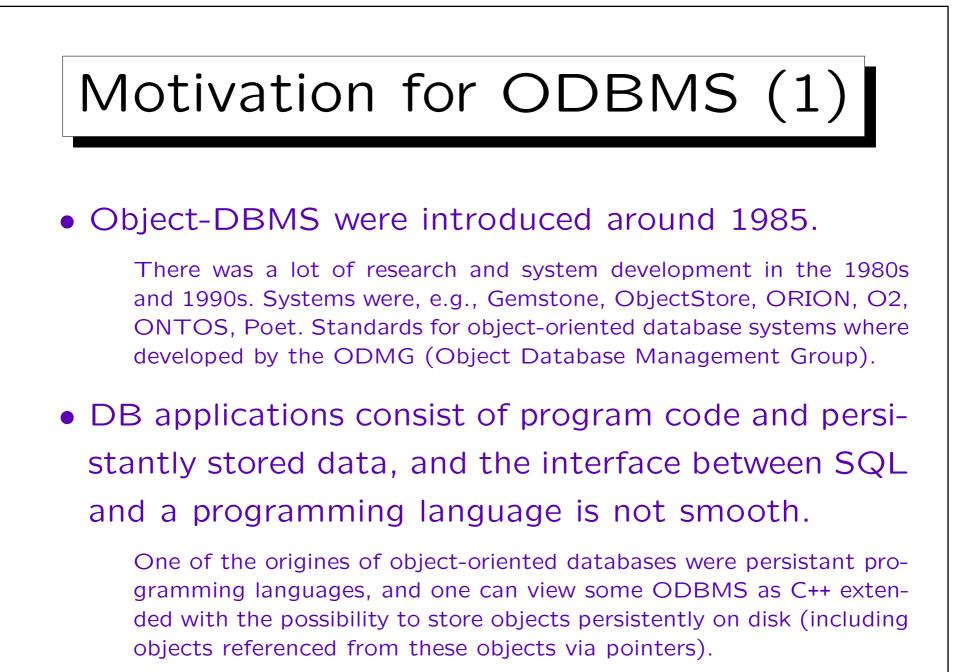


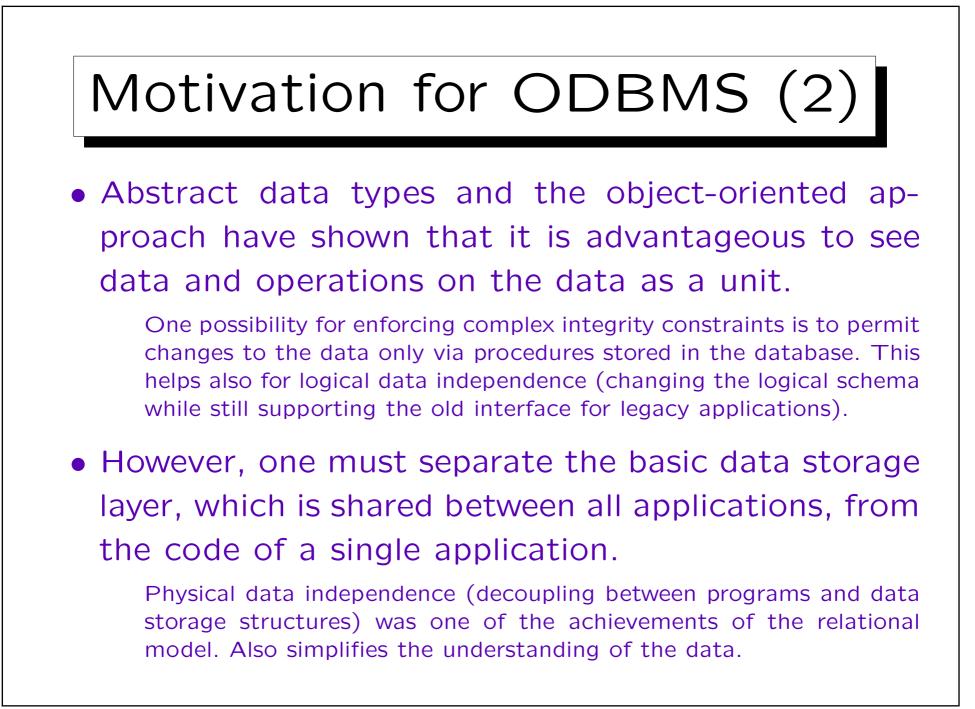
After completing this chapter, you should be able to:

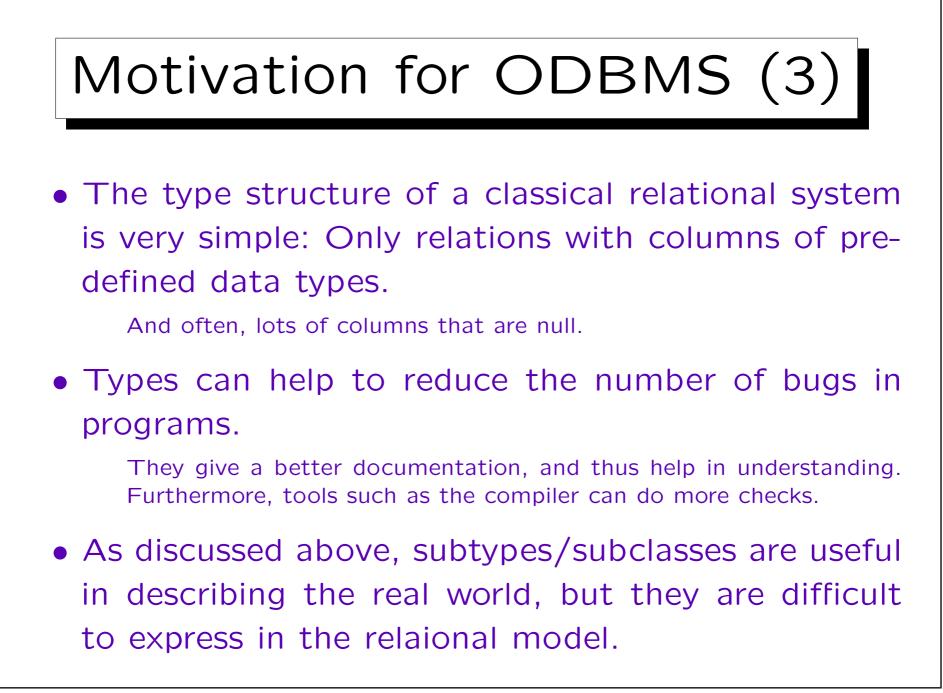
- name some features of object-relational databases,
- define an object-type in Oracle,

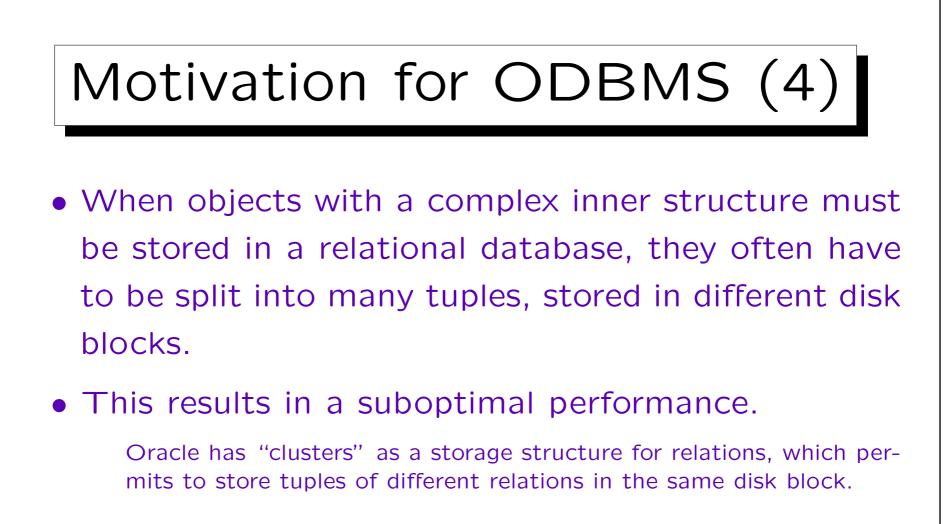


Stefan Brass: Datenbanken II A



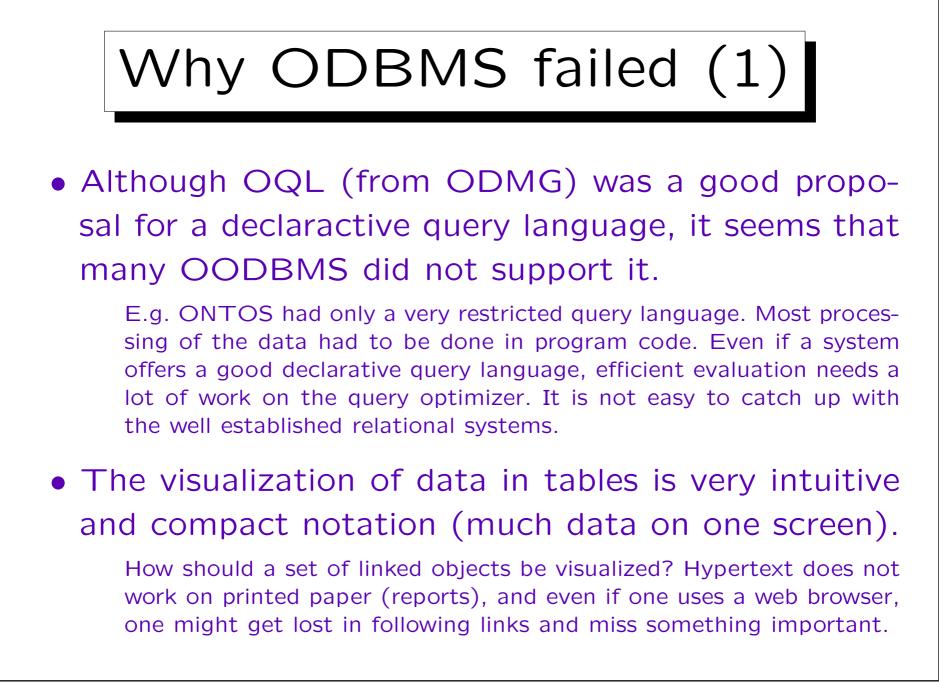


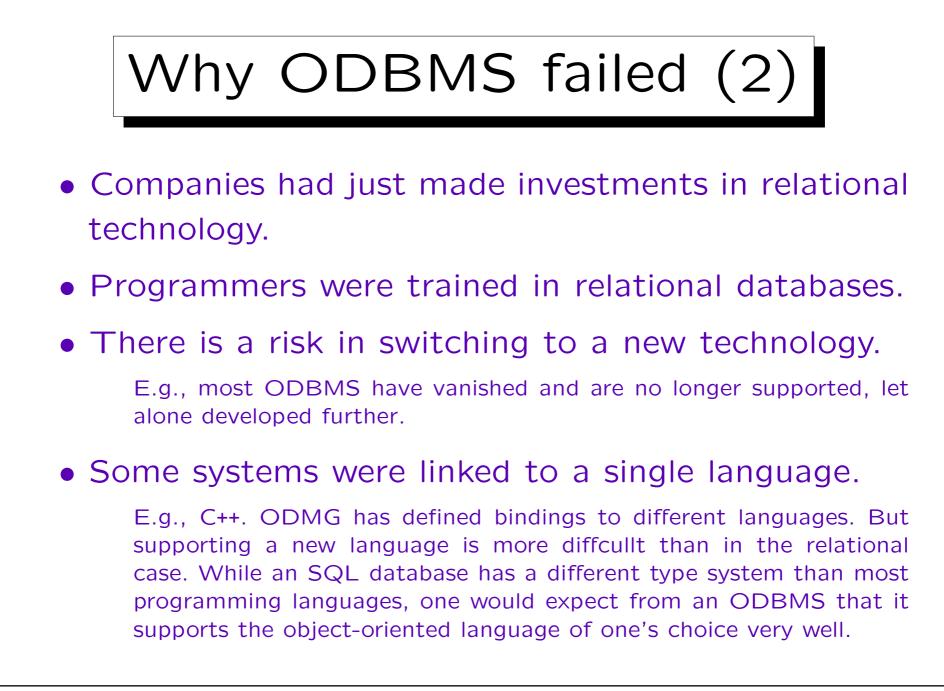




• Furthermore, one must write code for composing and decomposing an application object.

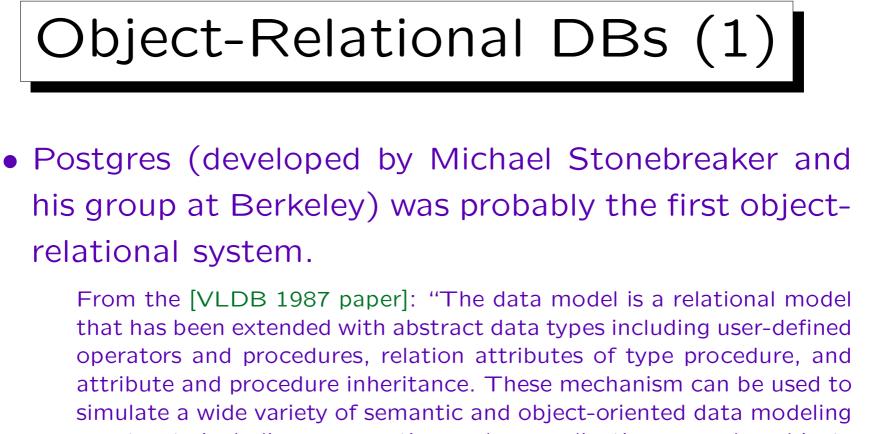
There are tools and libraries (frameworks) for this.





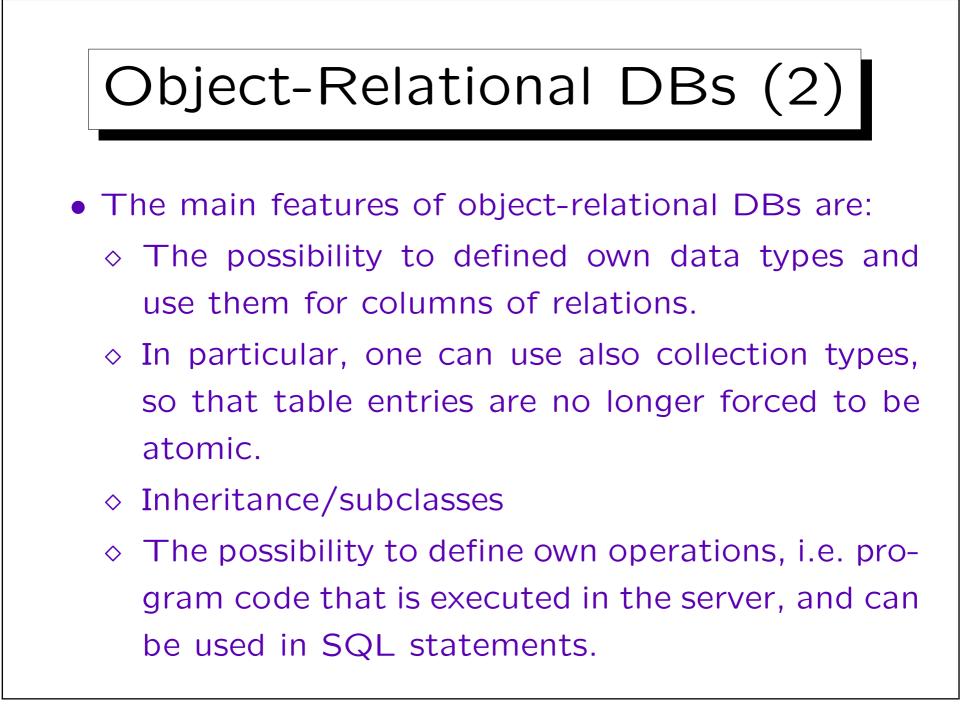


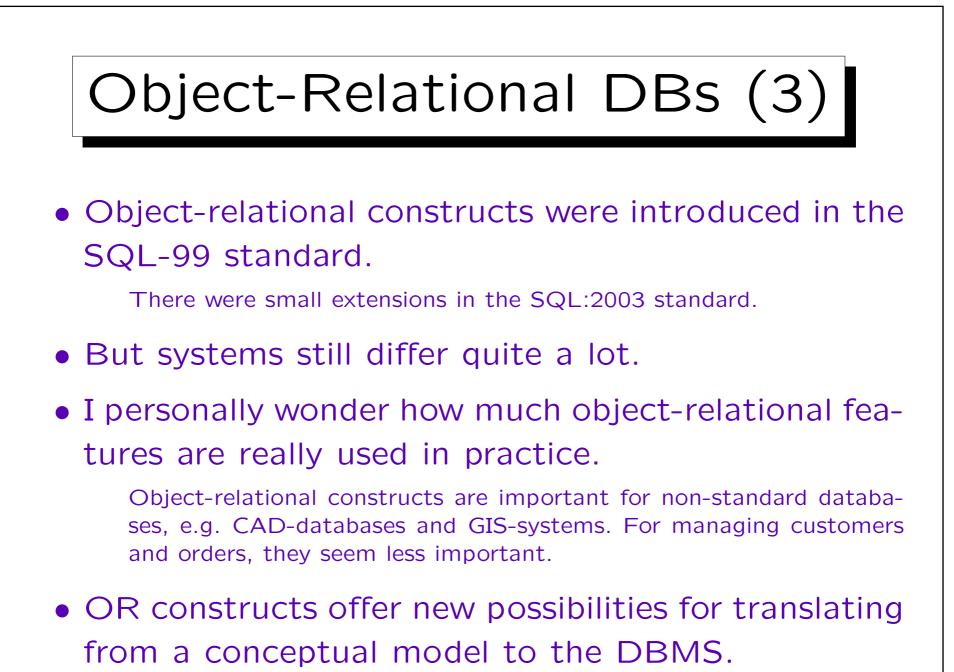
- Object/relational mappers (e.g., Hibernate, LINQ) offer a similar interface to the programmer, but are based on a standard relational database.
- The big relational vendors integrated some objectoriented features into their DBMS:
  - ♦ This gives object-relational database systems.
  - ◇ It permits a gradual change from an existing relational database to trying and using some objectoriented features where they seem useful.



constructs including aggregation and generalization, complex objects with shared subobjects, and attributes that reference tuples in other relations."

Object-relational systems have grown out of extensible database systems and systems that stored nested relations with more general collection types ("structurally object-oriented systems").

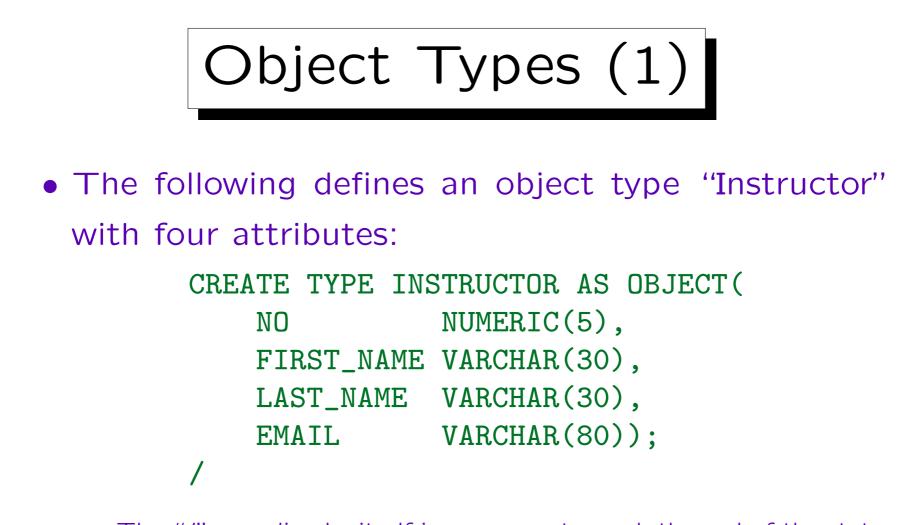




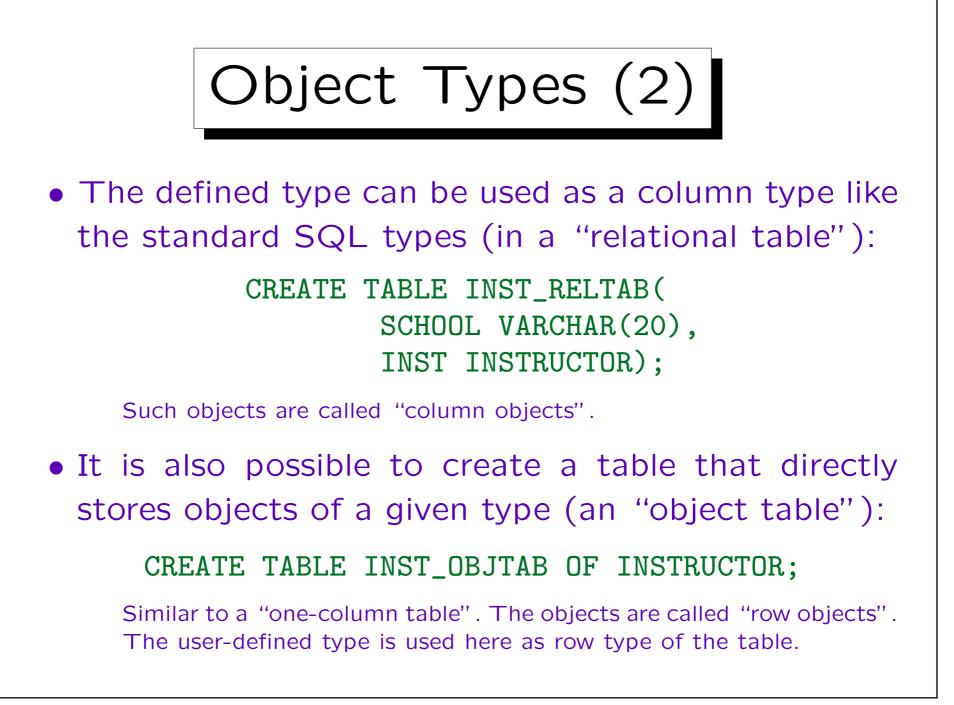


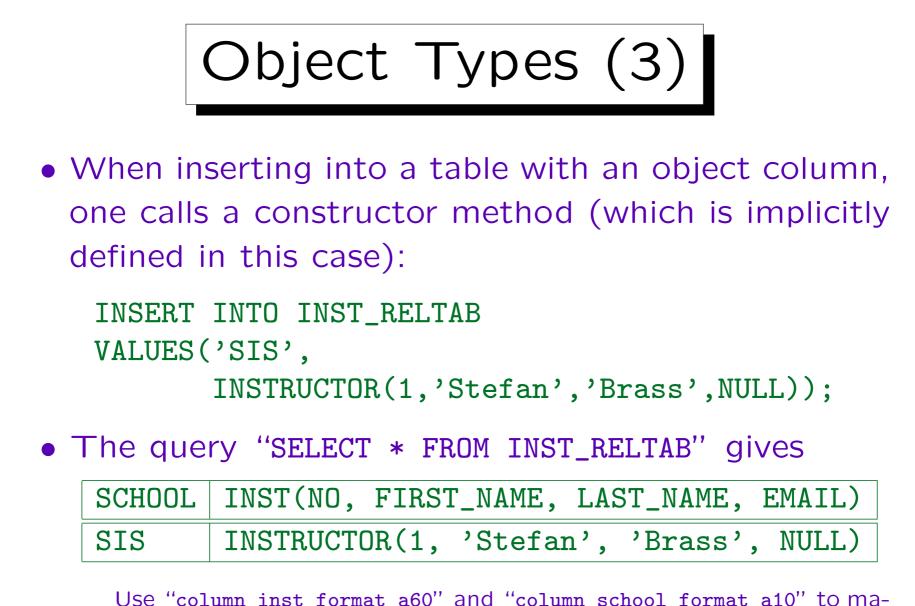
- 1. Introduction
- 2. Object Types
- 3. References
- 4. Subtypes
- 5. Collection Types

Stefan Brass: Datenbanken II A

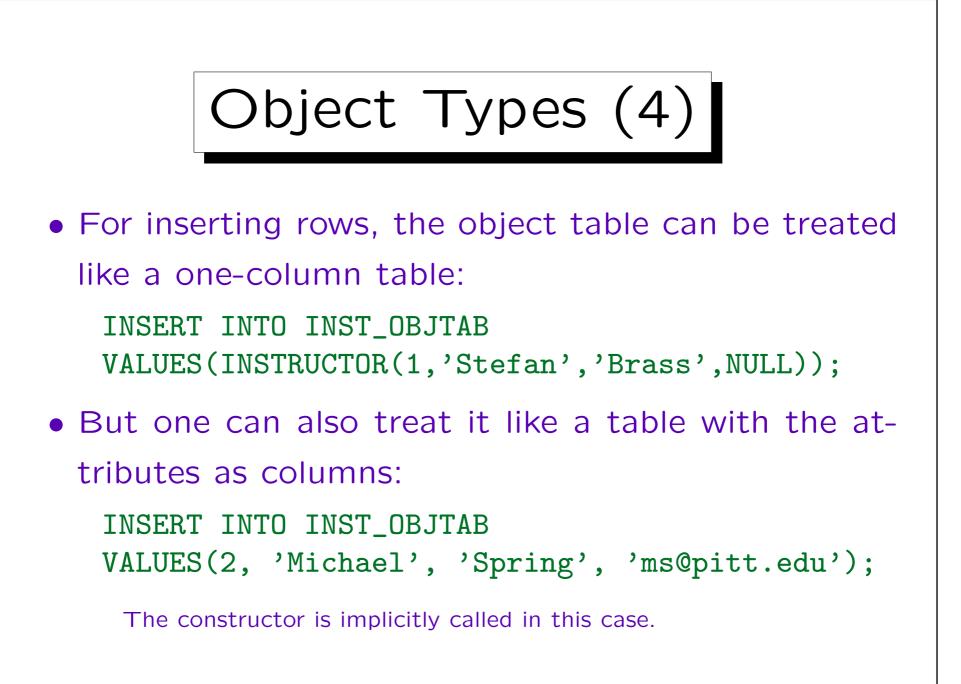


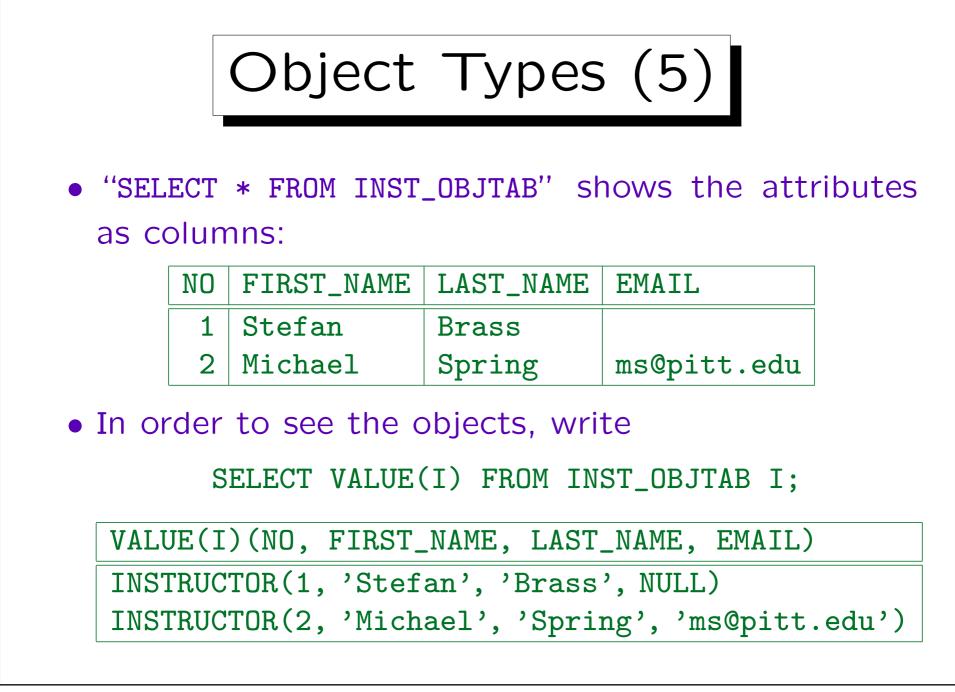
The "/" on a line by itself is necessary to mark the end of the statement. NOT NULL cannot be used. If there is a syntax error, SQL\*Plus will only print "Warning: Type created with compilation errors." Enter "show errors" in this case. Methods are discussed below.

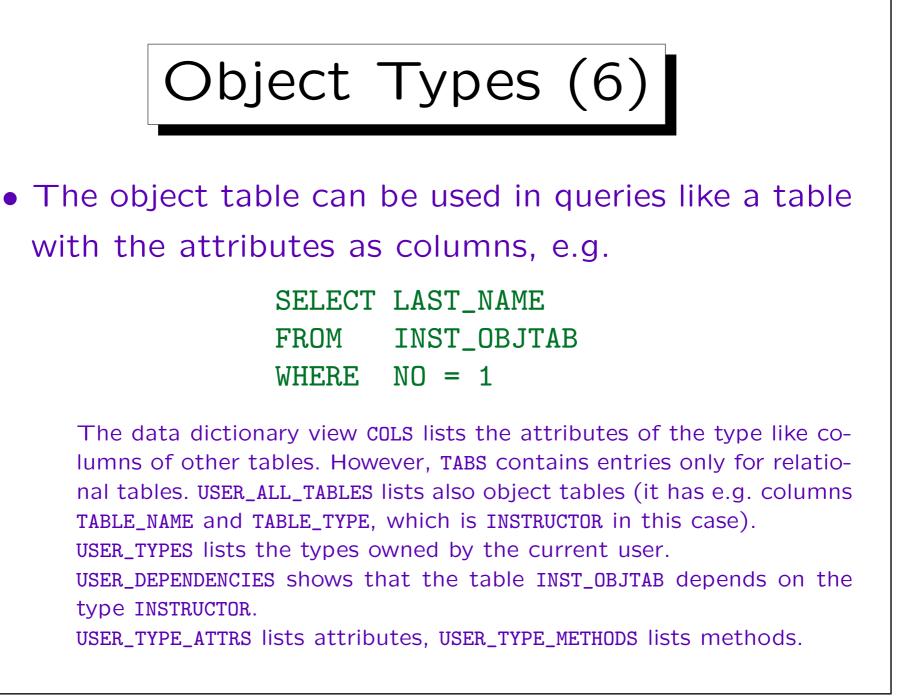


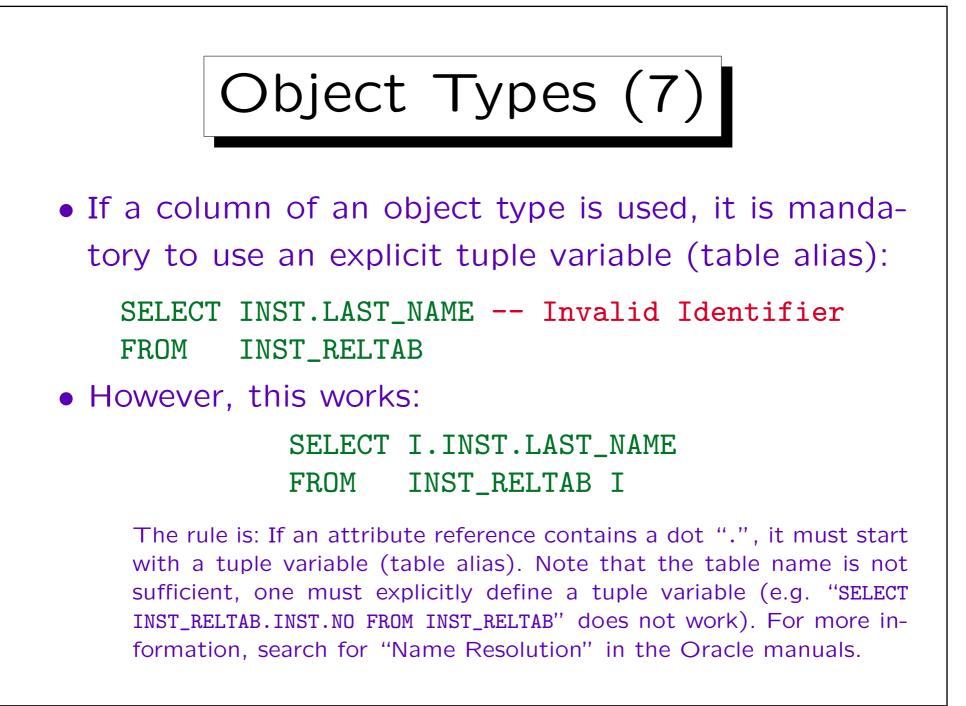


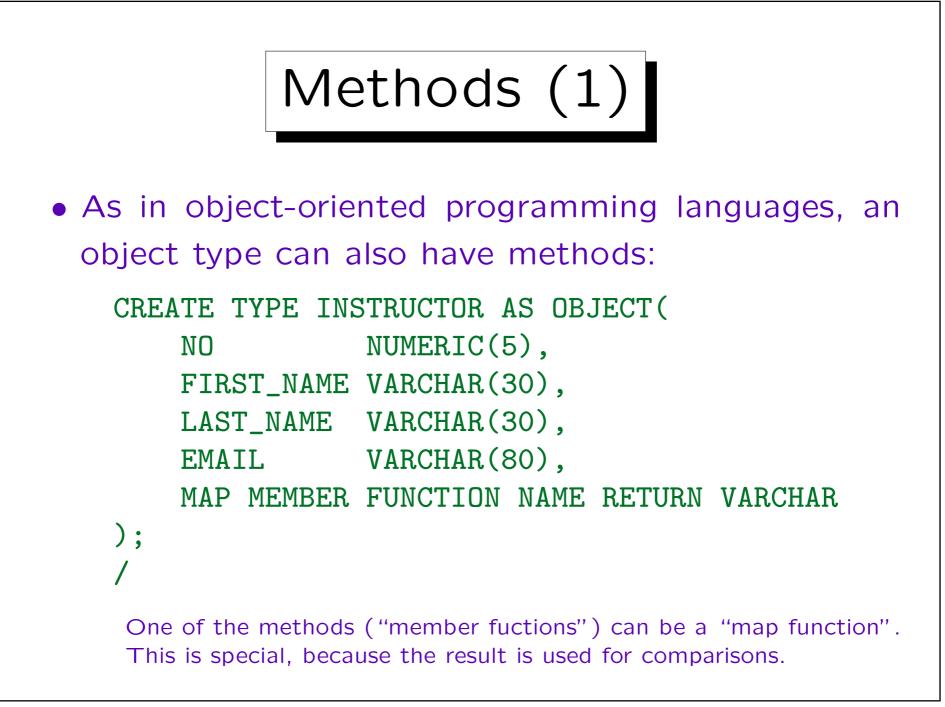
ke the display width of the columns smaller (so they fit into one line).

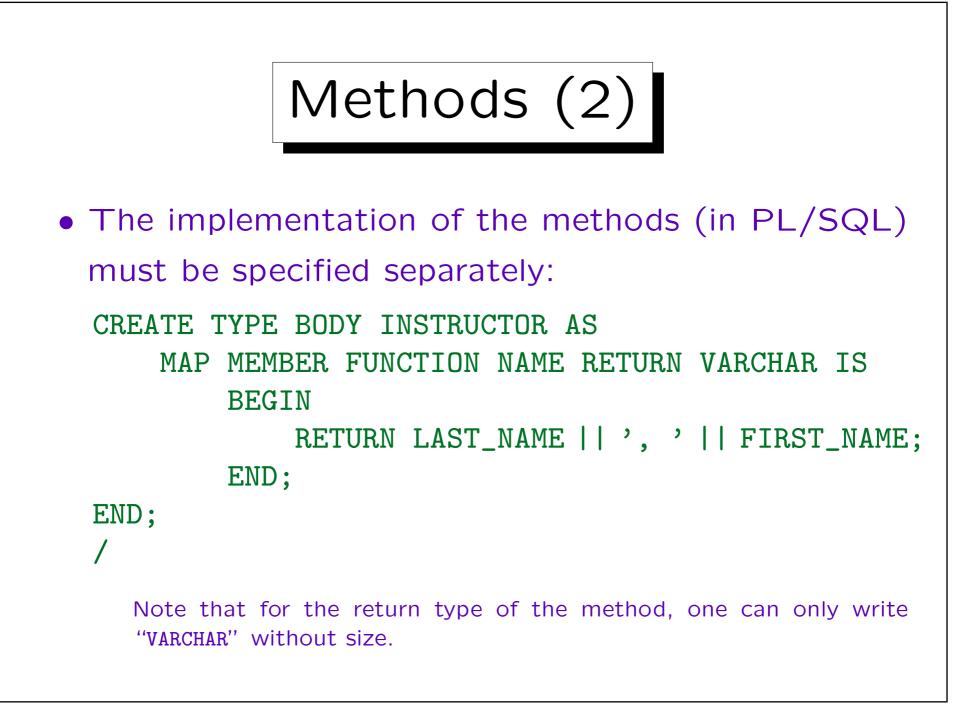


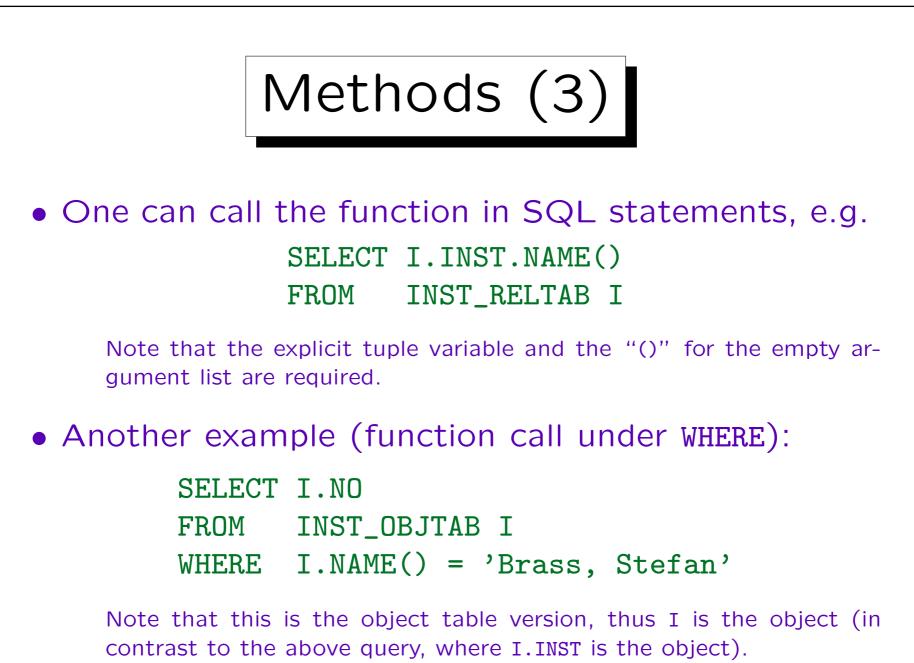














• Because NAME() was declared as the "MAP" member functions, objects are comparable (by mapping them to strings with this function):

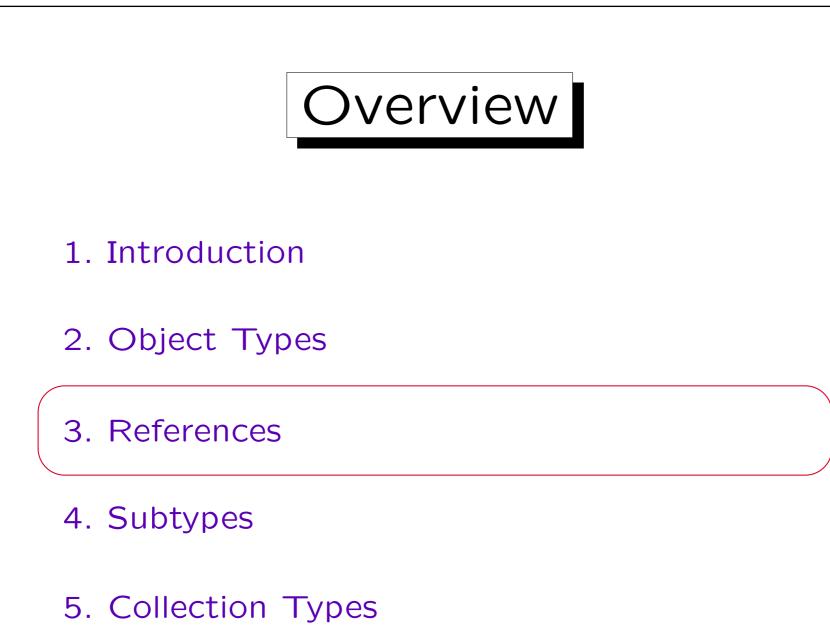
> SELECT I.SCHOOL, J.SCHOOL FROM INST\_RELTAB I, INST\_RELTAB J WHERE I.INST = J.INST

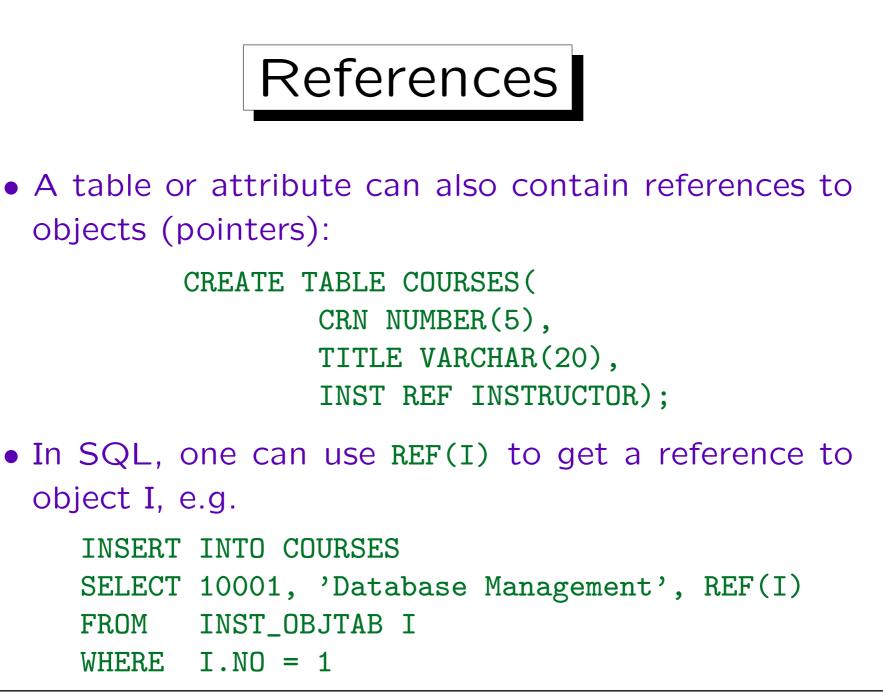
Objects will be considered identical if the have the same NAME() value — even if they differ in other attributes. It is probably a bad map function, if this can happen.

• One can also use <, > and ORDER BY for a class with map function.

Methods (5)

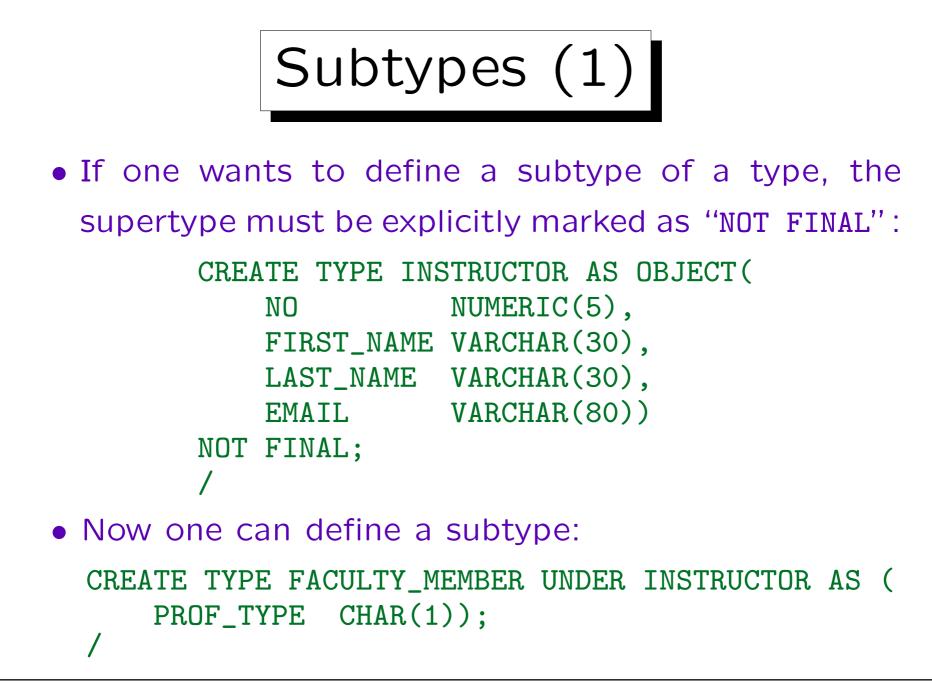
- It is also possible to define procedures in classes which can have side effects.
- Also static methods are available.
- One can also define own constructors.
- It seems that everything in the CREATE TYPE is public, and one cannot define attributes in the private CREATE TYPE BODY.







- 1. Introduction
- 2. Object Types
- 3. References
- 4. Subtypes
- 5. Collection Types



Subtypes (2)

- An object of the subtype FACULTY\_MEMBER can be inserted into a table with the declared type INSTRUCTOR: INSERT INTO INST\_OBJTAB VALUES ( FACULTY\_MEMBER(3, 'Paul', 'Munro', NULL, 'A'))
- "SELECT \* FROM INST\_OBJTAB" shows only the attributes of the type declared for the table:

NO	FIRST_NAME	LAST_NAME	EMAIL
1	Stefan	Brass	
2	Michael	Spring	ms@pitt.edu
3	Paul	Munro	

Subtypes (3)

• The full objects are shown as follows:

SELECT VALUE(I) FROM INST\_OBJTAB I;

VALUE(I)(NO, FIRST\_NAME, LAST\_NAME, EMAIL)

INSTRUCTOR(1, 'Stefan', 'Brass', NULL)
INSTRUCTOR(2, 'Michael', 'Spring', 'ms@pitt.edu')
FACULTY\_MEMBER(3, 'Paul', 'Munro', NULL, 'A')



- 1. Introduction
- 2. Object Types
- 3. References
- 4. Subtypes
- 5. Collection Types

Stefan Brass: Datenbanken II A