# QUESTIONNAIRE ON THE OBJECT-RELATIONAL IMPENDANCE MISMATCH

**Name:**

**Name:**

**Question 1.** Consider the concepts of Complex Concept (and all their related concepts, such as Type Classes, Extensibility, etc.) and Rich Type System. Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 2.** Consider the concepts of Object Identity and PK (include in the discussion the concepts of reference and FK). Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 3.** Consider how each paradigm access data. That is, high-level programming languages (typically computationally complete) Vs. SQL and the proposed extensions (i.e., constraints enforcement, updatable views, interfaces, etc.). Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 4.** Consider the position of both paradigms with regard to (code) encapsulation. Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 5.** Consider the concepts persistency, concurrency and recovery. Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 6.** Consider the concepts of Inheritance in both paradigms. Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 7.** Consider the concept of open system and the position of each paradigm. Are they equivalent, compatible or they represent a conflict between both approaches? Justify your answer

**Question 8.** Why do you think object-oriented databases did not succeed? In other words, what was the main drawback of object-oriented databases?

**Question 9.** What are the standard object-oriented features introduced by object-relational databases to smooth the impedance mismatch with object-oriented programming languages?

**Question 10**. Consider a Java application that defines tones of classes and stores its objects in a PostgreSQL by means of JDBC. Consider the same application but now running DB4o (or any other object-oriented database supporting Java) in the backend. When executing the mappings to store the Java objects into the database, could you name two OO features whose mapping to PostgreSQL (i.e., the JDBC code lines executed) will clearly perform worse than its counterpart mapping to DB4o? Justify your answer