PAPER CODE NO. **COMP302**	EXAMINER : Dr C. Ambühl DEPARTMENT : Computer Science Tel. No. 54264

THE UNIVERSITY
of LIVERPOOL

JANUARY 2007 EXAMINATIONS

Bachelor of Arts : Year 3
Bachelor of Engineering : Year 3
Bachelor of Science : Year 3
Master of Engineering : Year 4
No qualification aimed for : Year 1

Advanced Database Technology

TIME ALLOWED : 2.5 hours

INSTRUCTIONS TO CANDIDATES

Attempt all questions in Section A (60 marks in total).
Attempt **TWO** questions from Section B only (40 marks in total).

If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

SECTION A

Attempt ALL six questions from this section. Each question is worth 10 marks

1. Explain the ACID properties of transactions. **(10 marks)**

2. Translate the following relational schema to ODL. **(10 marks)**

   Person(PID, FName, LName, Street, City)
   Primary Key: PID

   IsMotherOf(MotherID, ChildID)
   Primary Key: (MotherID, ChildID) Foreign key: MotherID for Person(PID)
   Foreign key: ChildID for Person(PID)

3. Explain the two phase commit protocol used in distributed databases. **(10 marks)**

4. Name and describe the main differences between Relational Database Systems and Object Relational Database Systems. **(10 marks)**

5. Describe the difference between well-formed and validated XML documents. **(10 marks)**

6. Explain what the Document Object Model (DOM) is used for in the context of XML. **(10 marks)**

SECTION B

Attempt TWO questions from this section. Each question is worth 20 marks. Credit will be given for the best 2 answers only.

1. Consider the following ODL design:

```
class Ship
  (extent Ships)
{
  attibute name;
  attribute  enum Shiptype {sailing, steam, diesel} type
  attribute integer launched;
}
```

(a) Write a query in OQL that returns the names of all sailing ships launched after 1910. **(10 marks)**

(b) Write a query in OQL which returns the set of years between 1880 and 1900 in which more than 100 sailing ships have been launched. **(10 marks)**

2. Write a XML file that stores information about the following order:

   John Smith orders three books to be shipped to A. Anderson, Forest Road 34, Liverpool. The first book is a copy of ”Database systems: The complete book”, international edition, for £30. Additionally, he orders two copies of ”Database System Concepts”, for £50. The ID for the order is 65756.

   Make sure that your XML file could be **successfully validated** against the XML-schema on the next page. **(20 marks)**

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- definition of simple elements -->
  <xs:element name="orderperson" type="xs:string"/>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="address" type="xs:string"/>
  <xs:element name="city" type="xs:string"/>
  <xs:element name="country" type="xs:string"/>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="note" type="xs:string"/>
  <xs:element name="quantity" type="xs:positiveInteger"/>
  <xs:element name="price" type="xs:decimal"/>

  <!-- definition of attributes -->
  <xs:attribute name="orderid" type="xs:string"/>

  <!-- definition of complex elements -->
  <xs:element name="shipto">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="name"/>
        <xs:element ref="address"/>
        <xs:element ref="city"/>
        <xs:element ref="country"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="item">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="title"/>
        <xs:element ref="note" minOccurs="0"/>
        <xs:element ref="quantity"/>
        <xs:element ref="price"/>
      </xs:sequence
    </xs:complexType
  </xs:element>

  <xs:element name="shiporder">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="orderperson"/>
        <xs:element ref="shipto"/>
        <xs:element ref="item" maxOccurs="unbounded"/
      </xs:sequence>
      <xs:attribute ref="orderid" use="required"/
    </xs:complexType
  </xs:elemen

</xs:schema>
```

3. Execute the recovery algorithm on the log file given below. Start by stating the transactions which have to be redone or undone, respectively. Then state the operations for undoing and redoing. **(20 marks)**

```
<T0 start>
<T0, A, 0, 10>
<T0 commit>
<T1 start>
<T1, B, 0, 10>
<checkpoint {T1}>
<T2 start>
<T2, C, 0, 10>
<T2, C, 10, 20>
<checkpoint {T1, T2}>
<T1, B, 10, 20>
<T3 start>
<T3, A, 10, 20>
<T1 commit>
<T4 start>
<T4, B,20,33>
<T3, D, 0, 10>
<T3 commit>
!! crash  !!
```