SQL AND NOSQL

MID TERM EXAM

1. Choice questions:
2. NoSQL databases is used mainly for handling large volumes of \_\_\_\_\_\_\_\_\_\_\_\_\_\_ data.
3. Unstructured
4. Structured
5. Semi-structured
6. All of the mentioned
7. The \_\_\_\_\_\_\_\_\_\_\_ operation, denoted by −, allows us to find tuples that are in one relation but are not in another.
8. Union
9. Set-difference
10. Difference
11. Intersection
12. In which of the following can many entity instances of one type be related to many entity instances of another type?
13. One-to-One Relationship
14. One-to-Many Relationship
15. Many-to-Many Relationship
16. Composite Relationship
17. Relational Algebra does not have
18. Selection operator
19. Projection operator
20. Aggregation operators
21. Division operator
22. Normal form which only includes indivisible values or single atomic values is classified as
23. Third normal form
24. First normal form
25. Second normal form
26. Fourth normal form
27. Which of the SQL statements is correct?

a) SELECT Username AND Password FROM Users

b) SELECT Username, Password FROM Users

c) SELECT Username, Password WHERE Username = 'user1'

d) None of these

1. A UNION query is which of the following?
2. Combines the output from no more than two queries and must include the same number of columns.
3. Combines the output from no more than two queries and does not include the same number of columns.
4. Combines the output from multiple queries and must include the same number of columns.
5. Combines the output from multiple queries and does not include the same number of columns.
6. Disadvantages of DTD are

(i) DTDs are not extensible

(ii)DTDs are not in to support for namespaces

(iii)There is no provision for inheritance from one DTDs to another

a) (i) is correct

b) (i),(ii) are correct

c) (ii),(iii) are correct

d) (i),(ii),(iii) are correct

1. Which of the following XML documents are well-formed?
2. <firstElement>some text goes here

<secondElement>another text goes here</secondElement>

</firstElement>

1. <firstElement>some text goes here</firstElement>

<secondElement> another text goes here</secondElement>

1. <firstElement>some text goes here

<secondElement> another text goes here</firstElement>

</secondElement>

1. </firstElement>some text goes here

</secondElement>another text goes here

<firstElement>

1. Why do we use exist method in Xquery?
2. To determine if the XML data contains a certain node
3. To examine the XML and return back a scalar value
4. To Shred the XML nodes of the XML data into relational columns
5. To search inside xml data types
6. Consider the following two tables:

Table Name: Employee

Attributes: Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Department

Table Name: Incentives

Attributes: Employee\_id, Incentive\_date, Incentive\_amount

Write SQLs for the following scenarios:

1. Get First\_Name from employee table in upper case

Answer: Select upper(First\_name) from Employee

1. Get unique DEPARTMENT from employee table

Answer: Select distinct(Department) from Employee

1. Select first 3 characters of FIRST\_NAME from EMPLOYEE

Answer: Select substring(First\_name,1,3) from Employee

1. Get length of FIRST\_NAME from employee table

Answer: Select len(First\_name) from Employee

1. Get FIRST\_NAME, Joining year, Joining Month and Joining Date from employee table

Answer: Select First\_name, substring (convert(varchar,Joining\_date,103),7,4) , SUBSTRING (convert(varchar,Joining\_date,100),1,3) , SUBSTRING (convert(varchar,Joining\_date,100),5,2) from Employee

1. Get all employee details from the employee table order by First\_Name Ascending and Salary descending

Answer: Select \* from Employee order by First\_name asc, Salary desc

1. Get employee details from employee table whose employee name are not “John” and “Roy”

Answer: Select \* from Employee where First\_ name not in ('John','Roy')

1. Get employee details from employee table whose Salary between 500000 and 800000

Answer: Select \* from Employee where Salary between 500000 and 800000

1. Get employee details from employee table whose joining month is “January”

Answer: Select \* from Employee where substring(convert(varchar,joining\_date,100),1,3)='Jan'

1. Get department, total salary with respect to a department from employee table order by total salary descending

Answer: Select Department,sum(Salary) TotalSalary from employee group by DEPARTMENT order by TotalSalary descending

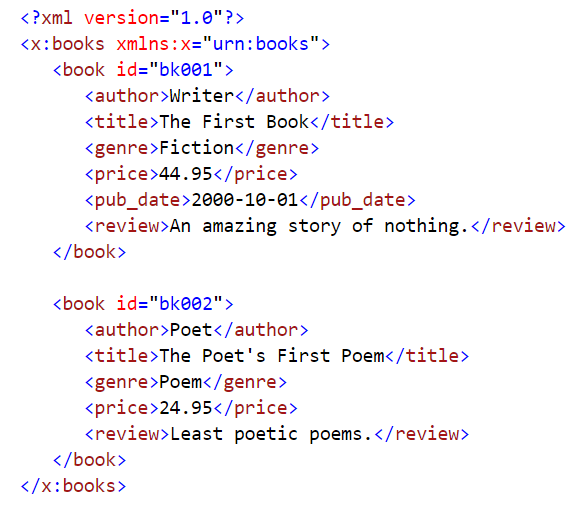
1. Write the DTD for the following xml file:



Answer:

<?xml version="1.0" encoding="UTF-8"?>  
<!ELEMENT DatabaseInventory (DatabaseName+)>  
<!ELEMENT DatabaseName ( GlobalDatabaseName  
 , OracleSID  
 , DatabaseDomain  
 , Administrator+  
 , DatabaseAttributes  
 , Comments)  
>  
<!ELEMENT GlobalDatabaseName (#PCDATA)>  
<!ELEMENT OracleSID (#PCDATA)>  
<!ELEMENT DatabaseDomain (#PCDATA)>  
<!ELEMENT Administrator (#PCDATA)>  
<!ELEMENT DatabaseAttributes EMPTY>  
<!ELEMENT Comments (#PCDATA)>  
  
<!ATTLIST Administrator EmailAlias CDATA #REQUIRED>  
<!ATTLIST Administrator Extension CDATA #IMPLIED>  
<!ATTLIST DatabaseAttributes Type (Production|Development|Testing) #REQUIRED>  
<!ATTLIST DatabaseAttributes Version (7|8|8i|9i) "9i">  
  
<!ENTITY AUTHOR "Jeffrey Hunter">  
<!ENTITY WEB "www.iDevelopment.info">  
<!ENTITY EMAIL "jhunter@iDevelopment.info">

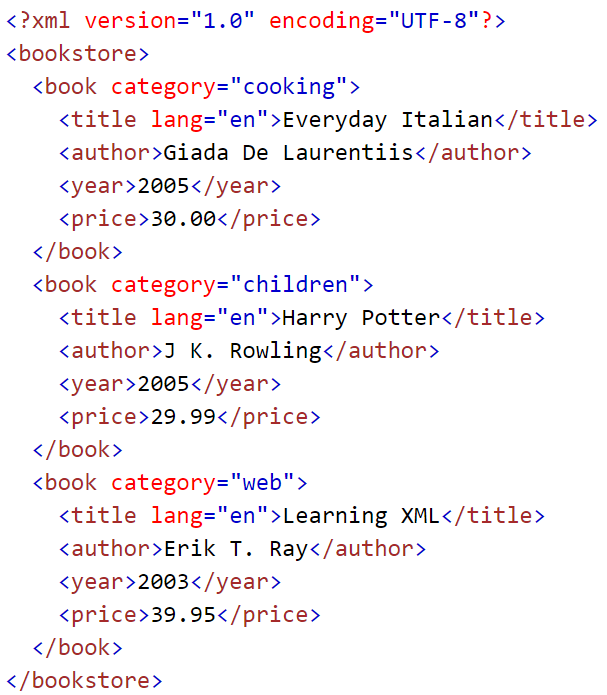
1. Write XML schema for the following XML file:



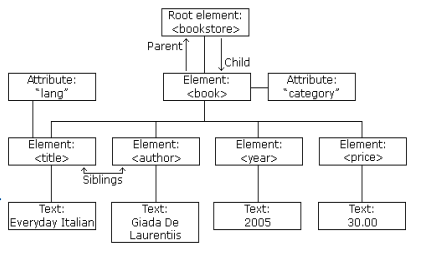
Answer:

<?xml version="1.0" encoding="UTF-8"?>  
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
 targetNamespace="urn:books"  
 xmlns:bks="urn:books">  
   
 <xsd:element name="books" type="bks:BooksForm"/>  
   
 <xsd:complexType name="BooksForm">  
 <xsd:sequence>  
 <xsd:element name="book"   
 type="bks:BookForm"   
 minOccurs="0"   
 maxOccurs="unbounded"/>  
 </xsd:sequence>  
 </xsd:complexType>  
   
 <xsd:complexType name="BookForm">  
 <xsd:sequence>  
 <xsd:element name="author" type="xsd:string"/>  
 <xsd:element name="title" type="xsd:string"/>  
 <xsd:element name="genre" type="xsd:string"/>  
 <xsd:element name="price" type="xsd:float" />  
 <xsd:element name="pub\_date" type="xsd:date" />  
 <xsd:element name="review" type="xsd:string"/>  
 </xsd:sequence>  
 <xsd:attribute name="id" type="xsd:string"/>  
 </xsd:complexType>  
</xsd:schema>

1. Write XML tree for the following XML file:



Answer:



1. For the xml below, answer the questions:

<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  
<book category="cooking">  
  <title lang="en">Everyday Italian</title>  
  <author>Giada De Laurentiis</author>  
  <year>2005</year>  
  <price>30.00</price>  
</book>  
  
<book category="children">  
  <title lang="en">Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>  
  
<book category="web">  
  <title lang="en">XQuery Kick Start</title>  
  <author>James McGovern</author>  
  <author>Per Bothner</author>  
  <author>Kurt Cagle</author>  
  <author>James Linn</author>  
  <author>Vaidyanathan Nagarajan</author>  
  <year>2003</year>  
  <price>49.99</price>  
</book>  
  
<book category="web">  
  <title lang="en">Learning XML</title>  
  <author>Erik T. Ray</author>  
  <year>2003</year>  
  <price>39.95</price>  
</book>  
  
</bookstore>

Write XPaths for the following scenarios:

1. Select the first book element that is the child of the bookstore element

**Answer**: /bookstore/book[1]

1. Selects the last but one book element that is the child of the bookstore element

**Answer**: /bookstore/book[last()-1]

1. Select the first two book elements that are children of the bookstore element

**Answer**: /bookstore/book[position()<3]

1. Select all the title elements that have a "lang" attribute with a value of "en"

**Answer**: //title[@lang='en']

1. Select all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

**Answer**: /bookstore/book[price>35.00]/title

1. General SQL and NoSQL questions:
2. What is the difference between JOIN and UNION?

**Answer**:

The main purpose of both of them is to combine the data into one result that can provide meaningful observations. Join combine the columns from different tables. We use join to produce the data based on logical relationship from multiple tables.

Union combine the data based on rows. Union combine the data from two or more quires into a single result. The columns should have same datatype.

1. What are aggregate and scalar functions? Give examples

**Answer:**

**Aggregate Function:**

These are mostly used on column and the output is single value. Some example of them are AVG(), Count() and First().

**Scalar Function:**

These functions are based on user input and they also return single output. Examples are UCASE(), NOW(), FORMAT().

1. What is the difference between NoSQL & Mysql DBs’?

**Answer:**

1. MySQL databases have fixed or static schema whereas NoSQL databases have dynamic schema.
2. MySQL databases are vertically scalable whereas NoSQL databases are horizontally scalable.
3. MySQL databases displays data in table form so they are called as table based databases whereas NoSQL databases displays data as Key value pair, JSON, XML etc.
4. MySQL databases are categorized as relation databases whereas NoSQL are categorized as non-relational or distributed database.
5. MySQL databases are best for complex queries whereas NoSQL are best for simple queries.
6. When should a NoSQL database be used instead of a relational database?

**Answer:**

When it comes to enterprise where you have to follow the ACID on a transaction, we go for Relational databases. Speed might be slow as we can perform complex quires in relational databases.

NoSQL is good for speed and scaling. When you deal with high volume of data, NoSQL is best as querying or operating on that data will have good performance. They are best when we have to serve the concurrency. In traditional RDBMS, we always see a lock on a table or record when we perform some action on it and thus it won’t allow other to modify the table until the complete transaction executes successfully or fail.

NoSQL databases are good for space management. This is one of the best advantage as they have ability of flexible data storage. If we have to change the database structure in RDBMS, it will be a time-consuming process.

NoSQL are decentralized in nature, They are best when the business want to move their systems to cloud.