SQL AND NOSQL

EXAM

PARTH NAIK (naikpa)

Answers:

1. Multiple Choice Questions:
2. d) All of the mentioned
3. b) Set difference
4. c) Many-to-Many Relationship
5. d) Division Operator
6. b) First normal form
7. b) SELECT Username, Password FROM Users (If we ignore the absence of semicolon at the end)
8. c) Combines the output from multiple queries and must include the same number of columns.
9. c) ii), iii) are correct.
10. a) If the 2 elements are within a tag then b) is also correct.
11. a) To determine if the XML data contains a certain node.
12. SQL queries
13. SELECT UPPER(First\_name) AS FIRST\_NAME

FROM Employee;

1. SELECT DISTINCT Department

FROM Employee;

1. SELECT SUBSTRING(First\_name,1,3) AS FIRST\_THREE

FROM Employee;

1. SELECT LENGTH(First\_name) AS NameLength

FROM Employee;

1. SELECT First\_name, YEAR(Joining\_date) AS Joining\_year, MONTH(Joining\_date) AS Joining\_month, Joining\_date

FROM Employee;

1. SELECT \*

FROM Employee

ORDER BY First\_name ASC, Salary DESC;

1. SELECT \*

FROM Employee

WHERE NOT First\_Name=”John” OR NOT First\_name=”Roy”;

1. SELECT \*

FROM Employee

WHERE Salary >= 500000 AND Salary <= 800000;

1. SELECT \*

FROM Employee

WHERE MONTH(Joining\_date) = 1;

1. SELECT Department, SUM(Salary) AS TotalSalary

FROM Employee

GROUP BY Department

ORDER BY TotalSalary DESC;

1. DTD

Inside DatabaseInventory.dtd

<!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 CDATA #REQUIRED>

<!ATTLIST DatabaseAttributes Version CDATA #REQUIRED>

1. XML Schema

<?xml version=”1.0”?>

<xs:schema xmlns:xs=”<http://www.w3.org/2001/XMLSchema>“ targetNamespace=”urn:books”>

<xs:element name=”books”>

<xs:complexType>

<xs:sequence>

<xs:element name=”author” type=”xs:string”/>

<xs:element name=”title” type=”xs:string”/>

<xs:element name=”genre” type=”xs:string”/>

<xs:element name=”price” type=”xs:decimal”/>

<xs:element name=”pub\_date” type=”xs:date” minOccurs=”0” maxOccurs=”1”/>

<xs:element name=”review” type=”xs:string”/>

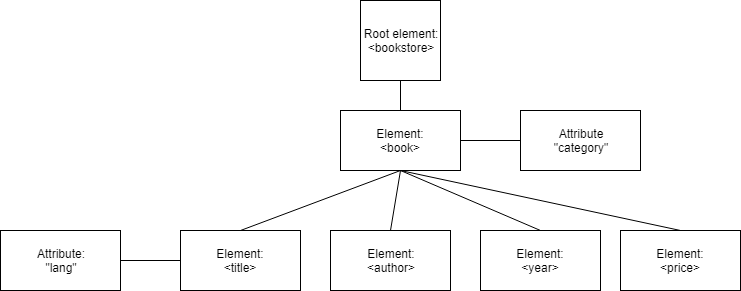
</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

1. XML tree



1. XPath
2. /bookstore/book[1]
3. /bookstore/book[last()-1]
4. /bookstore/book[position()<3]
5. //title[@lang=’en’]
6. /bookstore/book[price>35]/title
7. General SQL and NoSQL questions
8. Difference between JOIN and UNION

|  |  |
| --- | --- |
| JOIN | UNION |
| 1. JOIN clause in SQL is used to combine attributes of tuples of different relations having common features or attributes. | 1. The UNION clause in SQL is used to combine the tuples of relations in the query. |
| 1. JOIN is applicable when the involved relations have at least one common attribute. | 1. UNION is applicable when the involved relations have the same number of attributes and they belong to the same domain. |
| 1. INNER, OUTER, LEFT, RIGHT are its types. | 1. UNION and UNION ALL are its types. |
| 1. The length of the resultant tuples is more than the length of the tuples involved. | 1. The number of the resultant tuples is more as compared to the number of tuples present in each of the relations involved in the query. |

1. Aggregate and scalar functions

SQL has many built-in functions in order to manipulate and analyze data. These built-in functions can be broadly classified into:

* Aggregate functions
* Scalar functions

**Aggregate functions:**

Aggregate functions return a single value after performing calculations on a group of values. Some of the widely used aggregations functions are:

1. AVG()
2. COUNT()
3. FIRST()
4. LAST()
5. MAX()
6. MIN()
7. SUM()

Example of use of aggregate functions:

Consider a relation having employee information – Employee names, their departments and respective salaries.

If we want the average salary per department, we can write a query

SELECT Department, AVG(Salary) AS AverageSalary

FROM Employee

GROUP BY DEPARTMENT;

**Scalar functions:**

Scalar functions return a single value from an input value. Some of the commonly used scaler functions are:

1. UCASE()
2. LCASE()
3. MID()
4. LEN()
5. ROUND()
6. NOW()
7. FORMAT()

Example of use of scalar functions:

Let’s say that we want the accuracy of machine learning models stored in a relational table having the model name and accuracy score as the 2 columns rounded up to 2 decimal places. We can write a query as

SELECT ModelName, ROUND(Accuracy,2) AS RoundedAccuracy

FROM Models;

1. What is the difference between NoSQL and MySQL databases?

|  |  |  |
| --- | --- | --- |
| Parameter | SQL | NoSQL |
| Definition | SQL databases are primarily Relational Databases commonly abbreviated as RDBMS. | NoSQL databases are called as non-relational or distributed database. |
| Query Language | Structured Query Language | No declarative query language |
| Type | SQL databases are table-based databases. | NoSQL databases can be document-based, key-value pairs, graph databases. |
| Schema | SQL databases have a predefined schema. | NoSQL databases have a dynamic schema. |
| Ability to scale | SQL databases are vertically scalable. | NoSQL databases are horizontally scalable. |
| ACID vs BASE | ACID(Atomicity Consistency, Isolation and Durability) is a standard for RDBMS. | BASE(Basically Available, Soft state, Eventually Consistent) is a model of many NoSQL systems. |
| Best features | Cross-platform support, Secure and free. | Easy to use, high performance and flexibility. |
| Best opinion | When dynamic queries are required. | When scalability is required. |
| Examples | Oracle, Postgres and MS-SQL. | MongoDB, Redis, Neo4J, Cassandra, Hbase. |

1. When should a NoSQL database be used instead of a relational database?

Reasons to Use a NoSQL Database

To prevent the database from becoming a system-wide bottleneck, especially in high volume environments, NoSQL databases perform in a way that relational databases cannot.

The following features are driving the popularity of NoSQL databases like MongoDB, CouchDB, Cassandra, and HBase:

* **Storing large volumes of data without structure.** A NoSQL database doesn’t limit storable data types. Plus, you can add new types as business needs change.
* **Using cloud computing and storage.** Cloud-based storage is a great solution, but it requires data to be easily spread across multiple servers for scaling. Using affordable hardware on-site for testing and then for production in the cloud is what NoSQL databases are designed for.
* **Rapid development.** If you are developing using modern agile methodologies, a relational database will slow you down. A NoSQL database doesn’t require the level of preparation typically needed for relational databases.