# java.interview.preparation

This blog is to provide details for java interview preparation. Everybody is busy in day to day project work and most of times we are not able to spend time due to workload. I am trying to provide the consolidated details of java topics and sharing my java interview experiences.

Home   Concurrency   Design Pattern   Spring MVC   Spring   JMS   Data structure   Collection   Threading   Hibernate   Face 2 Fa

Interview Planning   Interview Questions   Memory Mgmt & GC   XML   Puzzles   DB   Java 1.5   Transaction   Inner Class   Cache

Database Interview Questions for Java Developers Part 2

Here are more DB interview question for java developer. Mostly I have been asked question preliminary on DB Index, Query optimisation, Standard query and DB functions. DB Interview Questions : Part -1.
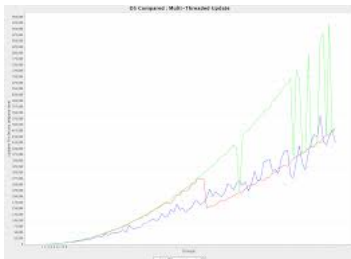
Q 1- How to Enable/Disable Indexes?
A - Disable Index
    ALTER INDEX [IndexName] ON TableName DISABLE GO
Enable Index
    ALTER INDEX [IndexName] ON TableName REBUILD

Q 2- **What is max number of columns in index?**
A- 32

Q 3- **How many maximum columns in partitions can be used?**
A - 16

Q 4- How to replace specific values from query result? i.e. how query return 'M' and 'F' can be converted into 'Male' and Female'.
A - This can be done using Oracle Decode function the syntax will be like

```
SELECT employee_name,
DECODE(type,'M', 'Male',
              'F', 'Female')type
FROM employee;
```

Q 5 - If the column of the table is having case sensitive data like 'Abc' and 'ABC' and we want to do case sensitive search on table. How we can improve performance of case sensitive search?
A- In this case we can define case sensitive index on table so that each search is restricted to particular value compare to full table scan. You can force all your data to be the same case by using UPPER() or LOWER():
  select * from my_table where upper(column_1) = upper('my_string');
or
  select * from my_table where lower(column_1) = lower('my_string');

If column_1 is not indexed on upper(column_1) or lower(column_1), as appropriate, this may force a full table scan. In order to avoid this you can create a function-based index.

create index my_index on my_table (lower(column_1));

If you're using LIKE then you have to concatenate a % around the string you're searching for.

select * from my_table where lower(column_1) = lower('my_string') || '%';

Q 5 - What are Synonyms?
A Synonyms give you the ability to provide alternate names for database objects. You can alias object names; for example, using the Employee table as Emp. You can also shorten names. This is especially useful when dealing with three and four part names; for example, shortening server.database.owner.object to object.

Q 6 - What is the Difference between Update Lock and Exclusive Lock?
A -When Exclusive Lock is on any process, no other lock can be placed on that row or table. Every other process have to wait till Exclusive Lock completes its tasks.
Update Lock is a type of Exclusive Lock, except that it can be placed on the row which already has Shared Lock on it. Update Lock reads the data of the row which has the Shared Lock as soon as the Update Lock is ready to change the data it converts itself to the Exclusive Lock.

Q 7 -What is a Surrogate Key?
A - A surrogate key is a substitution for the natural primary key. It is just a unique identifier or number for each row that can be used for the primary key to the table. The only requirement for a surrogate primary key is that it should be unique for each row in the table. It is useful because the natural

primary key can change and this makes updates more difficult. Surrogated keys are always integer or numeric.

A 8 - Design the Database process which can handle 1 million requests at same time and What should be DB consideration?
A - All the major databases and operating systems will work in 1 million requests at same time; provided sufficient hardware with sufficient bandwidth to memory, disk, and network are provided.

    All databases are built for this kind of scenario -- that is, where you need to update lots of rows from different clients all the time. This is what indexes and primary keys are used for, and the database is optimized for updating in this manner. (i.e., UPDATE your_table SET your_column=your_value where your_key=12)
The hardware requirements are going to be your biggest issue, and I suspect you will need to think about quite a lot here, including:

- Failover (what happens when your main server fails?)

- Clustering (You may simply need to have more than one database server to handle your load)

- Processors (2? 4? 8? 16? Multi-core? Does the db provide multi-core support that is well optimized?)

- Memory (The faster the better, but your chipset must be able to handle the bandwidth as well)

- Disk (Faster I/O the better. eSATA/SATA or Fiber, etc.)

- Network (You'll need lots of bandwidth to handle this kind of traffic)

- Virtualization (Does it make sense to build this as real hardware? Why not as virtual servers in the cloud? Amazon / Rackspace / etc.?)

Thankfully a good majority of the scaling issues are handled either at the hardware or db level. That is, you can start your database now on a slow server and as traffic increases you can adjust accordingly with minimal changes to your code or your db structure. What will change is your hardware and your db configuration. Most database servers will support failover/clustering with little to no change to your existing code or structures. (But be sure to research possible costs and the efficiency thereof first)
Good discussion on same issue.

Q 9 - Remove duplicates from table?
A:

```sql
DELETE FROM TableName
WHERE  ID NOT IN (SELECT MAX(ID)
                  FROM   TableName
                  GROUP  BY Column1,
                            Column2,
                            Column3
                  HAVING MAX(ID) IS NOT NULL)
```

Q 10 : Finding N'th Maximum salary SQL query?

```sql
// Using Sub query                                                          SELECT *
FROM Employee Emp1
WHERE (N-1) = (
              SELECT COUNT(DISTINCT(Emp2.Salary))
              FROM Employee Emp2
              WHERE Emp2.Salary > Emp1.Salary
              )

//Another way to get 2'nd maximum salary                        Select max(Salary) From Employee e where
                            e.sal < ( select max(sal) from employee );
```

Q 11 - Database Query Optimisation tips with JDBC:
A : Use prepared statement pooling
Database supports prepared statement pooling for pooled connections, as discussed in the JDBC 3.0 specification, through the TimesTenObservableConnectionDS class. Note that statement pooling is transparent to an application. Use of the PreparedStatement object, including preparing and closing the statement, is no different.

Use arrays of parameters for batch execution
You can improve performance by using groups, referred to as *batches*, of statement executions, calling the addBatch() and executeBatch() methods forStatement or PreparedStatement objects.
A batch can consist of a set of INSERT, UPDATE, DELETE, or MERGE statements. Statements that return result sets, such as SELECT statements, are not allowed in a batch. A SQL statement is added to a batch by calling addBatch() on the statement object. The set of SQL statements associated with a batch are executed through the executeBatch() method.
For PreparedStatement objects, a batch consists of repeated executions of a statement using different input parameter values. For each set of input values, create the batch by using appropriate set*XXX*() calls followed by the addBatch() call. The batch is executed by the executeBatch() method.

```java
// turn off autocommit
conn.setAutoCommit(false);
Statement stmt = conn.createStatement();
stmt.addBatch("INSERT INTO employees VALUES (1000, 'Joe Jones')");
stmt.addBatch("INSERT INTO departments VALUES (260, 'Shoe')");
stmt.addBatch("INSERT INTO emp_dept VALUES (1000, 260)");
// submit a batch of update commands for execution
int[] updateCounts = stmt.executeBatch();
conn.commit ();
```

Bulk fetch rows of data
Oracle provides an extension that allows an application to fetch multiple rows of data. For applications that retrieve large amounts of data, fetching multiple rows can increase performance greatly. However, when using Read Committed isolation level, locks are held on all rows being retrieved until the application has received all the data, decreasing concurrency. For more information on this feature, see "Fetching multiple rows of data".
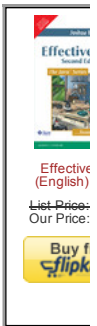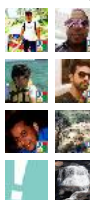
Use the ResultSet method getString() sparingly

Because Java strings are immutable, the ResultSet method getString() must allocate space for a new string in addition to translating the underlying C string to a Unicode string, making it a costly call.

In addition, you should not call getString() on primitive numeric types, like byte or int, unless it is absolutely necessary. It is much faster to call getInt()on an integer column, for example.

Avoid data type conversions

TimesTen instruction paths are so short that even small delays due to data conversion can cause a relatively large percentage increase in transaction time.

Use the appropriate get*XXX*() method on a ResultSet object for the data type of the data in the underlying database. For example, if the data type of the data is DOUBLE, to avoid data conversion in the JDBC driver you should call getDouble(). Similarly, use the appropriate set*XXX*() method on thePreparedStatement object for the input parameter in an SQL statement. For example, if you are inserting data into a CHAR column using aPreparedStatement, you should use setString().

Q 12: What should be strategy to minimise the impact on external system which is using our table's data for processing?

Answer : Best strategy should be using database views to minimize the impact on external system, If we are changing our table structure or columns and if we are able to populate same data using View then there should not be any impact on external system and they will not require any change.

Java Jobs

Opening for ".Net Developer" for one...
Hamilton, NJ

JAVA Entwickler (SENIOR)
München, Bayern, Germany
Michael Bailey Associates - Munich

Software Developer, C# .NET...
Oxford, Oxfordshire, United Kingdom
Haybrook IT Resourcing

Senior Java Developer
Limerick, Limerick, Ireland
Morgan McKinley

Hardware Development Security Engineer
Cambridge, Cambridgeshire, United...
LA International Computer Consultants...

POST A JOB >

POWERED BY JOBTHREAD

DB Jobs

Senior Programmer/Analyst
Cedarburg, WI
Systems Search

BI/Reporting Lead
Dublin, Dublin, Ireland
Morgan McKinley

Test Engineers
Bentonville, AR
SAGE IT

Principal Full Stack Java Developer
Dublin, Dublin, Ireland
Harvey Nash Group

SAS Developer
Albany, NY

POST A JOB >

POWERED BY JOBTHREAD

Testing job

SharePoint Product Team Leader
Stansted, Essex, United Kingdom
Gamit

Python Developer
Dublin, Dublin, Ireland
Morgan McKinley

C# Developer - Aylesbury Bucks
Aylesbury, Buckinghamshire, United...
MAPP Ltd.

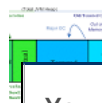Research & Development Test Lead
Galway, Galway, Ireland
Manpower UK

Web Search Evaluator –...
Anywhere in Australia, New South...
APPEN

POST A JOB >

POWERED BY JOBTHREAD

Understanding Spring MVC
Spring MVC is mostly used with Spring for any web application development. It is very powerful and nice layered architecture for flow and c...

Posted by Java Interview Questions

G+1 Recommend this on Google

Labels: database, investment bank, java investment bank, jdbc, Senior java developer interviews

No comments:

Post a Comment

Enter your comment...

Comment as: Unknown (Goc ▼

Sign out

Publish   Preview

☐ Notify me

Newer Post   Home   Older Post

Subscribe to: Post Comments (Atom)
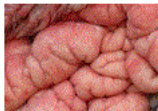
10 seconds test - what do you see?

◎ **Brain**   ◎ **Intestine**   ◎ **Fats**

yourtest-india

GOOGLE ANALYTICS

Picture Window template. Template images by Bim. Powered by Blogger.