Proactively Tuning Your Database Using the SQL Tuning Advisor

Purpose

This module describes how you can use Enterprise Manager and Automatic Workload Repository to proactively tune your database. Many of the analysis tools used by the Tuning Advisor are exposed, allowing the database administrator to perform reactive tuning as well.

Topics

This module will discuss the following topics:

- ☒ [Overview](#)
- ☒ [Prerequisites](#)
- ☒ [Viewing Database Waits](#)
- ☒ [Examining Top SQL for a Database Wait Class](#)
- ☒ [Tune a SQL Statement Using the SQL Tuning Advisor](#)
- ☒ [Reviewing SQL Execution Details for a SQL Statement](#)

Place the cursor on this icon to display all screenshots. You can also place the cursor on each icon to see only the screenshot associated with it.

Overview

[Back to Topic List](#)

What are the new SQL Tuning and Diagnostics features of Enterprise Manager?

The Automatic Workload Repository collects, processes, and maintains performance statistics for problem detection and self-tuning purposes. The Automatic Database Diagnostic Monitor (ADDM) reduces the amount of effort required to diagnosis and tune Oracle systems. The SQL Tuning Advisor feature allows a quick and efficient technique for optimizing SQL statements.

Performance diagnostic information can be viewed in Oracle Enterprise Manager screens after diagnostic monitoring has completed on the data. When performance problems are encountered, you can launch Oracle advisors to further define and correct the problems. For example, ADDM can identify high load SQL statements, then you can tune these statements with the SQL Tuning Advisor.

Proactive versus Reactive Database Tuning

With the new integrated and automatic tuning features of the Oracle Database 10 *g* , you can use Enterprise Manager to detect problems as they occur and devise solutions for the tuning problems. As the DBA, you would simply implement the recommendations. This is referred to as Proactive Tuning.

You can also use the same tools to perform your own analysis of database performance. You can create SQL Tuning

Tasks to resolve problems in the method best suited for your business. This is referred to as Reactive Tuning.

Prerequisites

Before starting this module, you should have:

1. Completed the [Configuring Linux for the Installation of Oracle Database 10g](#) lesson

2. Completed the [Installing the Oracle Database 10g on Linux](#) lesson

3. Download and unzip [perflab.tar](#) into your working directory (i.e. /home/oracle/wkdir)

Viewing Database Waits

You will first initiate several workload sessions. Then you will investigate the database workload. Perform the following:

1. Open a command line window, and run the following OS script:

   ```
   ./
   ```
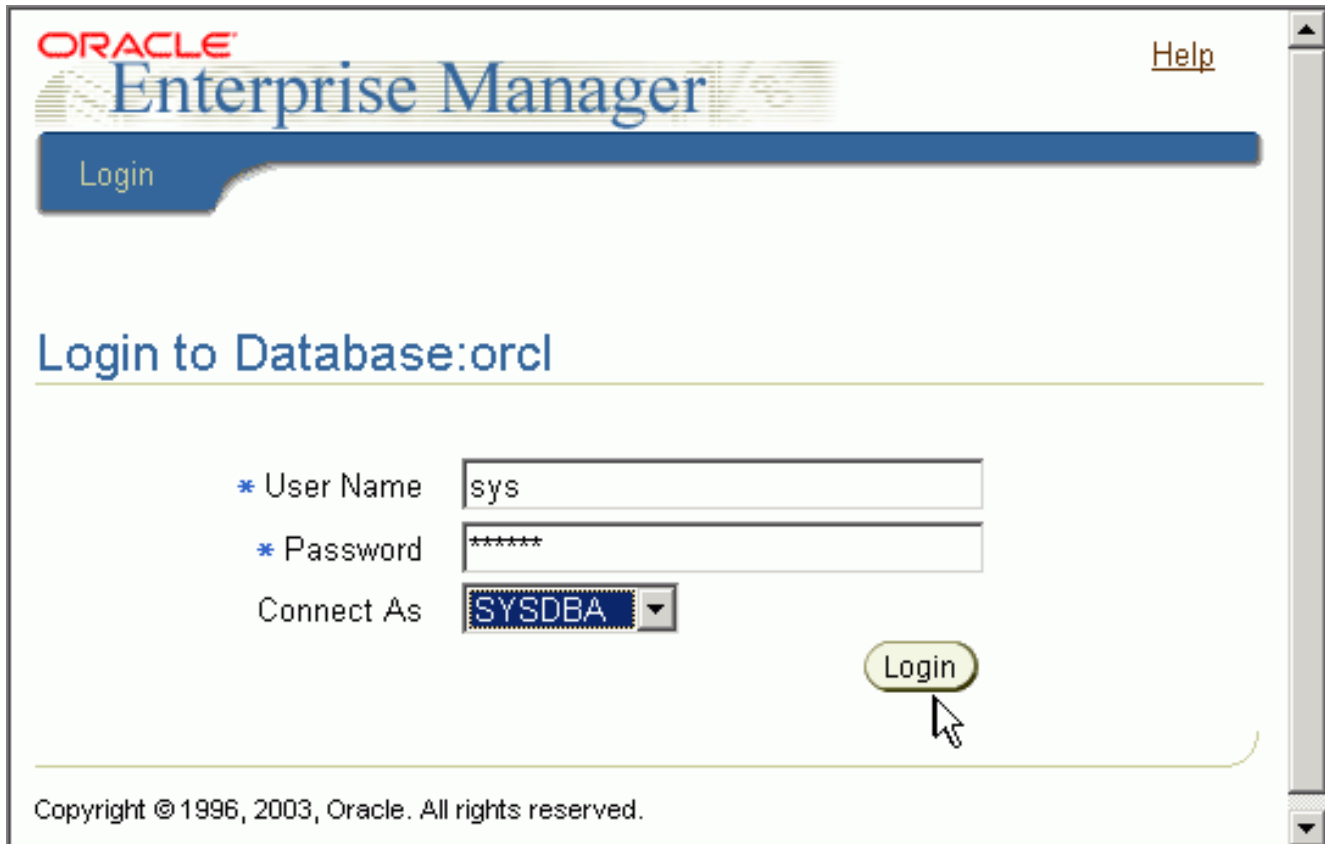
   ```
   setup_perflab.sh
   ```

   Note: This script will take approximately four minutes to run.

2. Open a browser and enter the following URL:

`http://<hostname>:5500/em`

Specify the User Name as **sys** and the Password **.** Choose **SYSDBA** from the Connect As drop down list, then click **Login.**

2. Select the **Administration** link.

**3.** In the section titled **Workload** , click on the **Automatic Workload Repository** link.
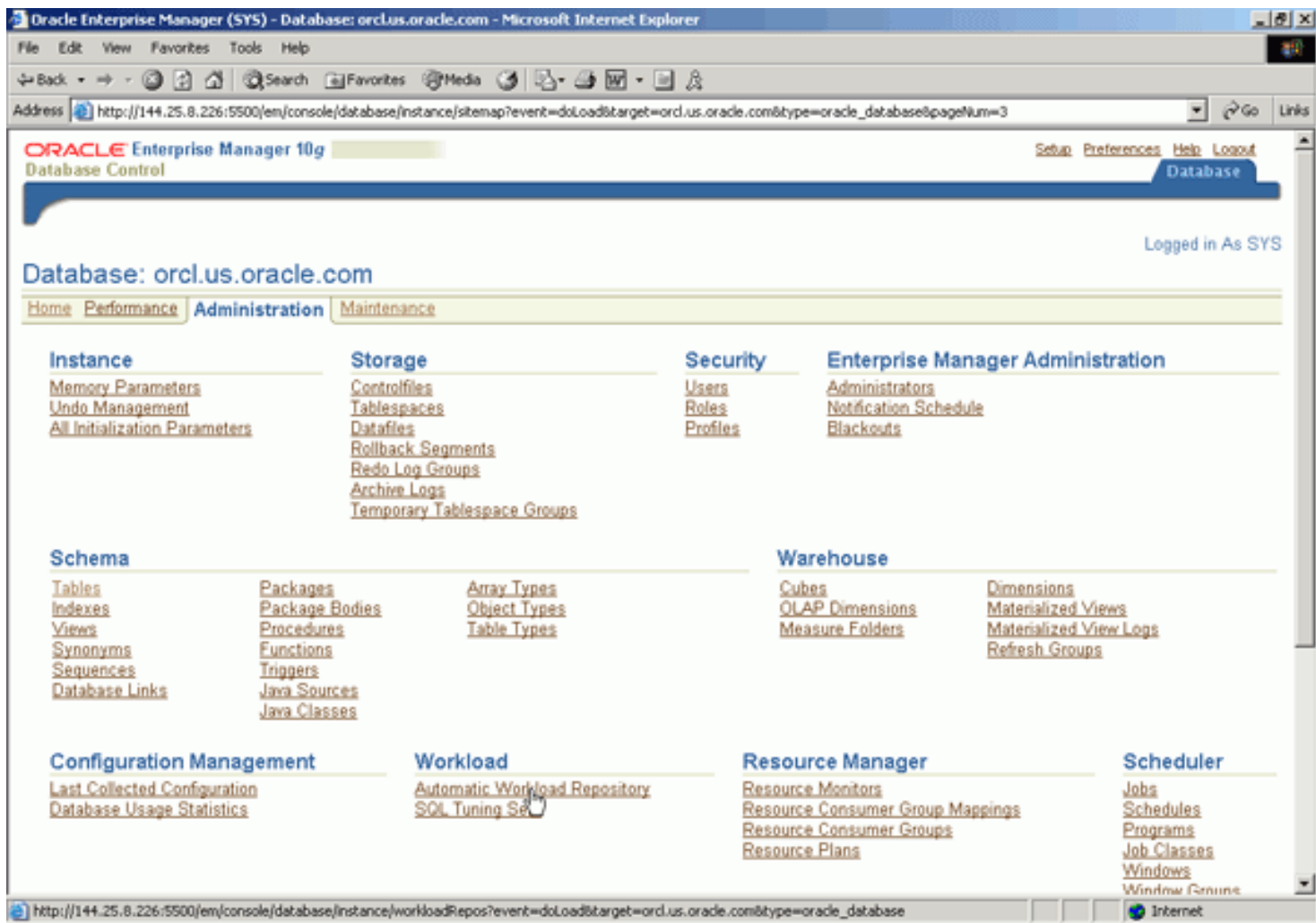
4. Determine how many snapshots have already been collected for this database. Look under **Snapshots** for the count and the time the last ADDM snapshot was taken. There should be at least three snapshots. Click on the **Database** breadcrumb.
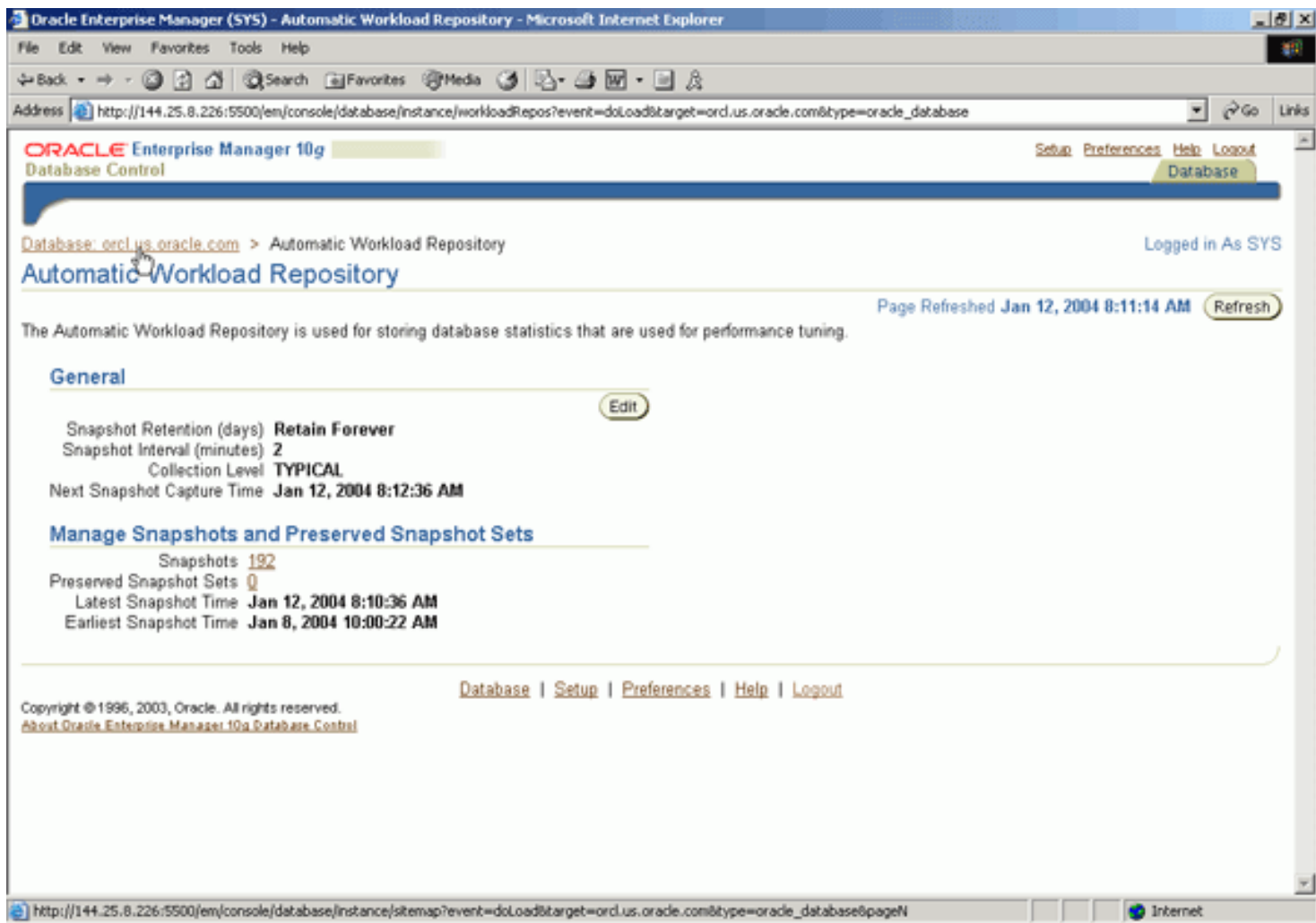
5. Click on the **Performance** link.
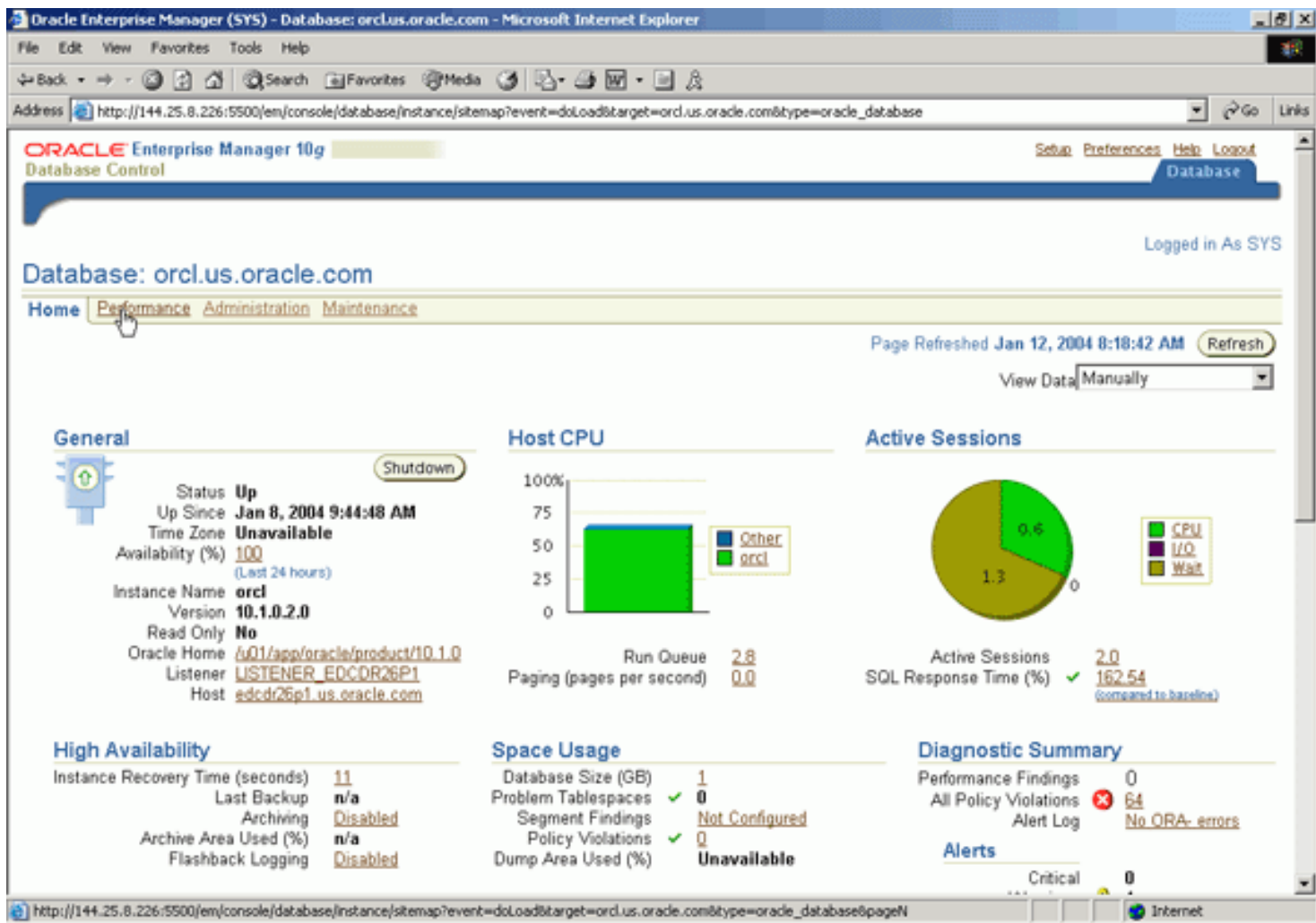
6. In the Performance Window, review to the **Sessions: Waiting and Working** graph. A chart representing the current workload of your database is shown. (It may take a minute for the chart to be populated with data) This chart is populated with data collected by the ADDM snapshots. To the side of the graph is the legend. Each legend entry is coded to a different color. You can determine quickly from the graph that the item with the largest time is yellow, or User I/O. Click on the **User I/O** link.
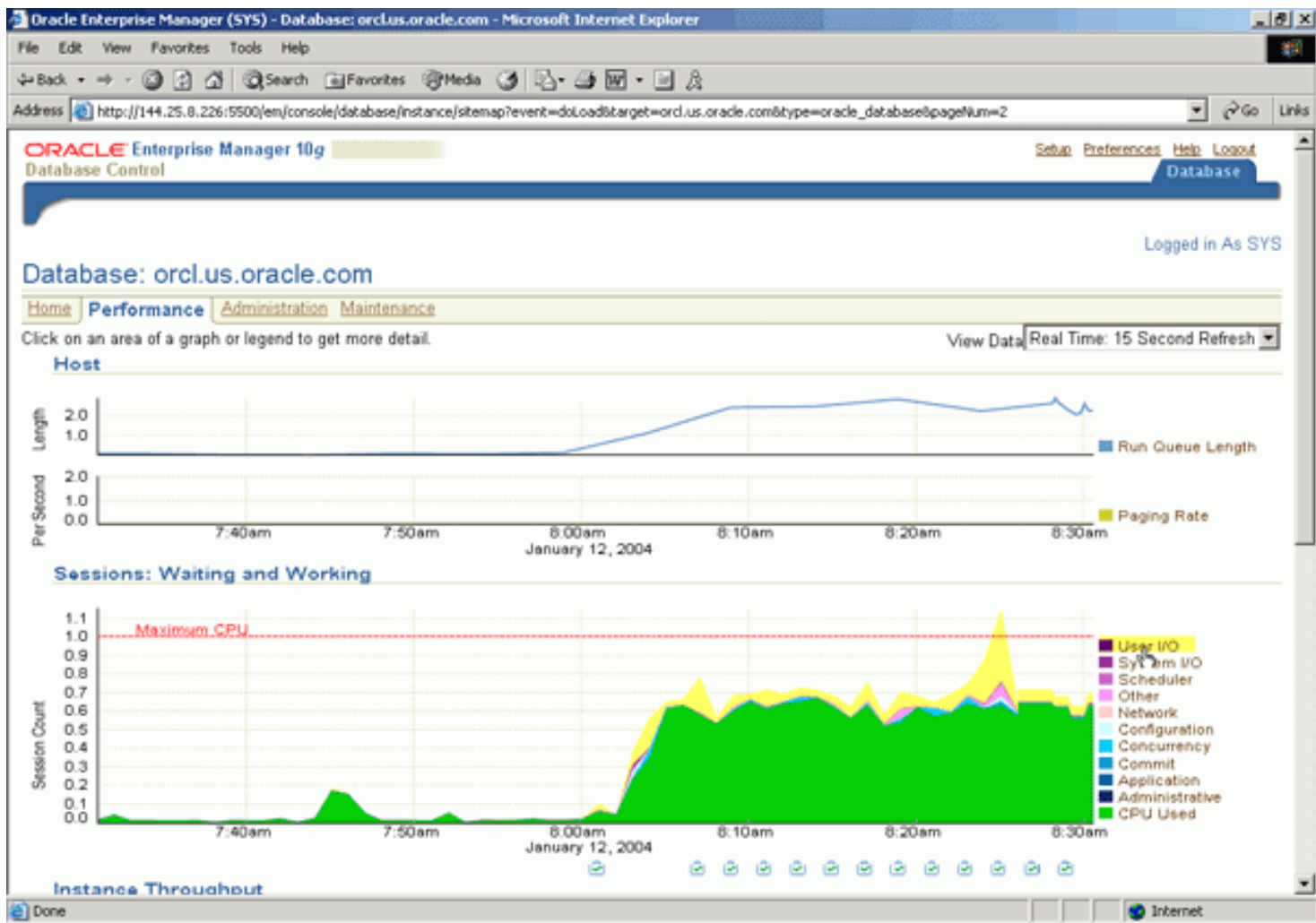
7. Below the Active Sessions Waiting: User I/O chart, there are two pie charts. Investigate the pie chart on the left, **Top Waiting SQL.** This shows that the overwhelming majority of waits 53% were caused by one SQL statement. Investigate the pie chart on the right, **Top Waiting Sessions.** This pie chart shows that the current top active sessions are waiting about the same percentage of time.
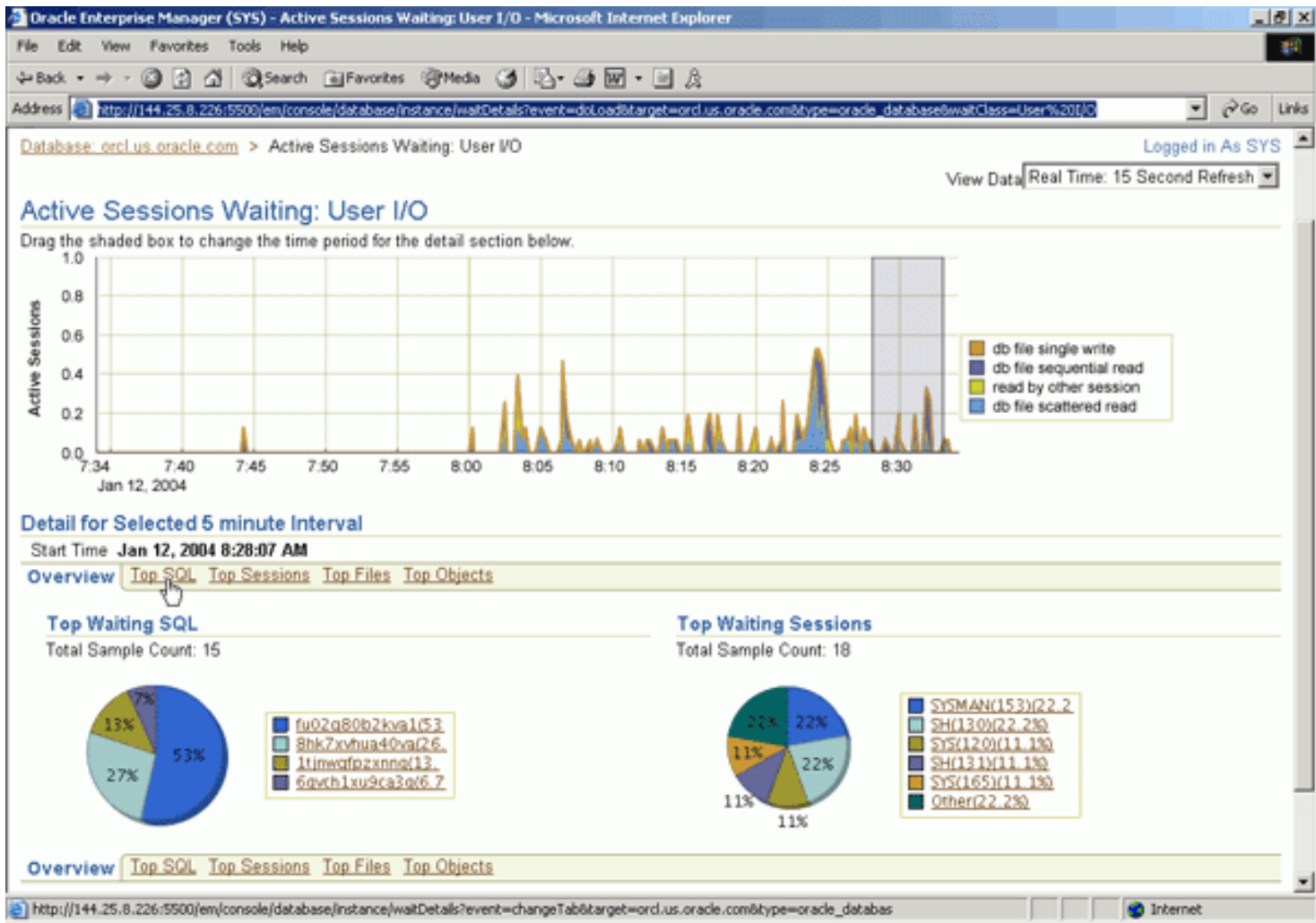
Examining Top SQL for a Database Wait Class

As was shown in the previous task, there is one SQL statement causing the majority of the database wait. In this task you will drill down to find the root cause. Perform the following:

1. From within the Active Sessions Waiting: User I/O page, click on the **Top SQL** tab in the middle of the page.
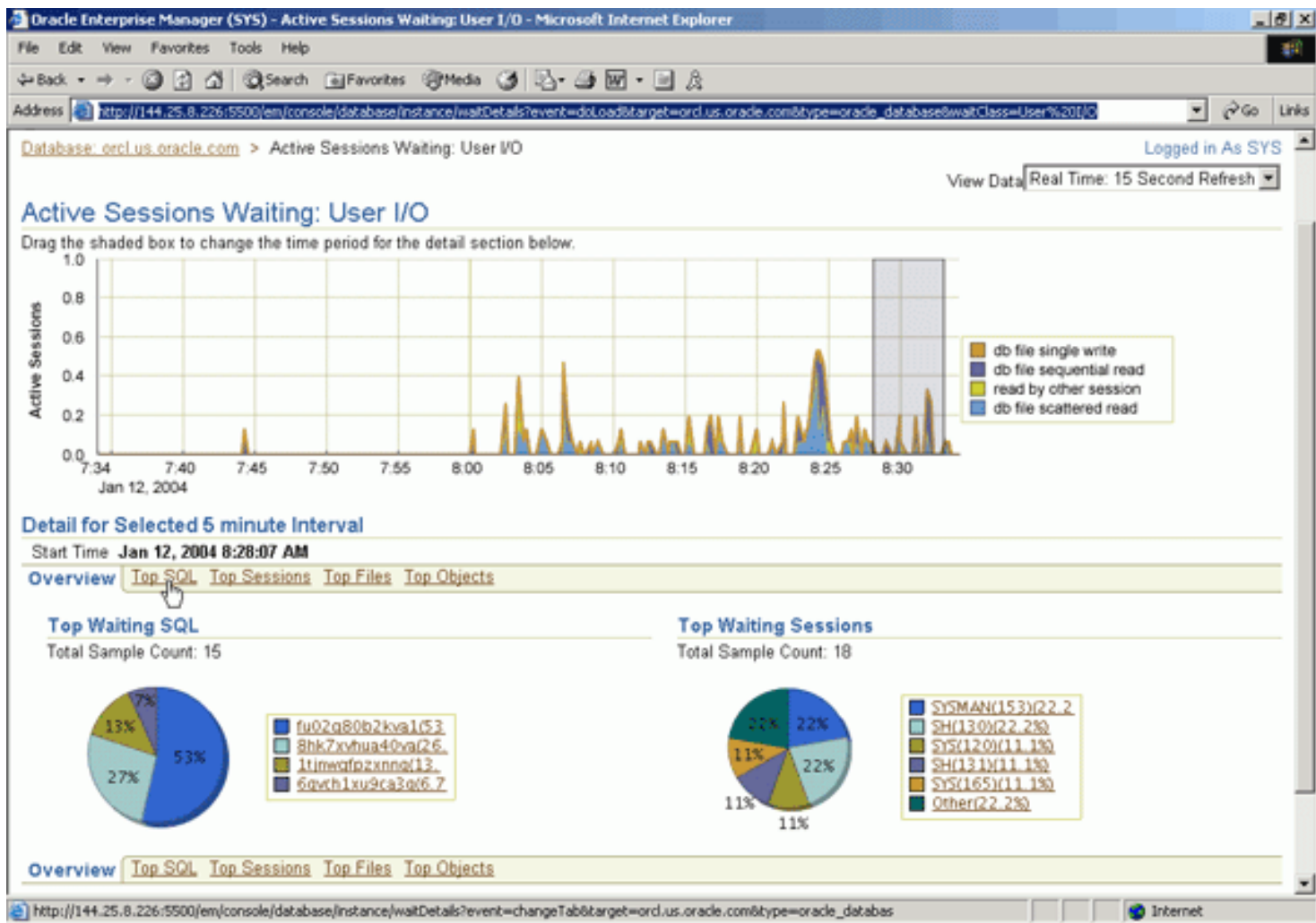
2. On the detail page that appears, view the **Wait Events for Top SQL** table, which is ordered by Activity (%). You can see the Top SQL statement spent most of its time on the activity 'db file scattered read'. Click on the SQL ID of the SQL statement with the highest percentage of activity.

3. The execution plan for this SQL statement is displayed. Click the **Current Statistics** tab.

4. The statistics for this SQL statement is displayed. Click the **Execution History** tab.

5. The statistical analysis chart for this SQL statement is displayed. The CPU and Elapsed Time chart shows the amount of CPU used by all executions of this SQL statement over a period of time. Select **Seconds Per Execution** and click on **Go** to display the time and resources used for each execution of this SQL statement.
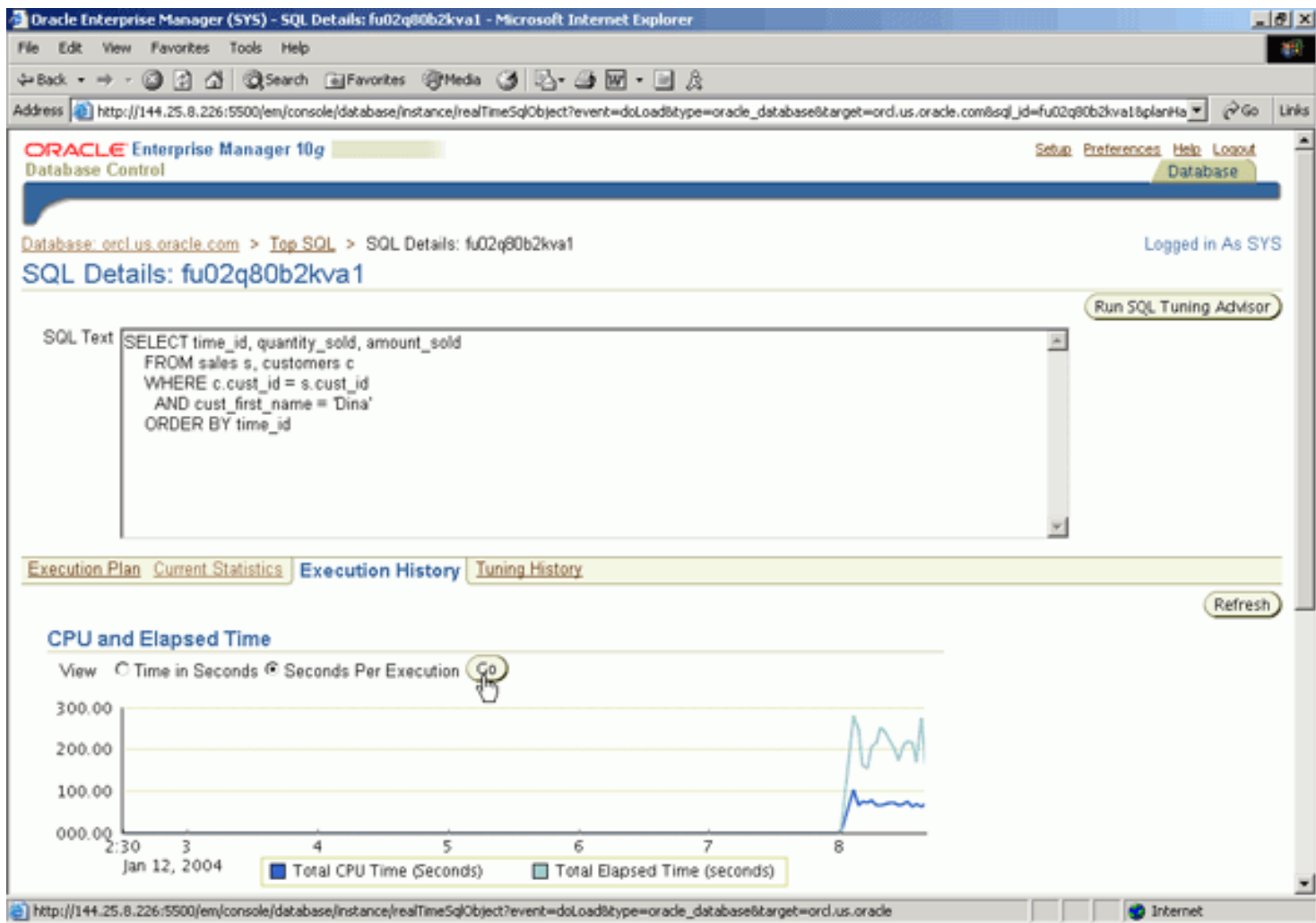
6. From the displayed charts, it can be determined that CPU resource usage is increasing, and the time it takes to execute this SQL statement is also increasing. Click the **Tuning History** tab.
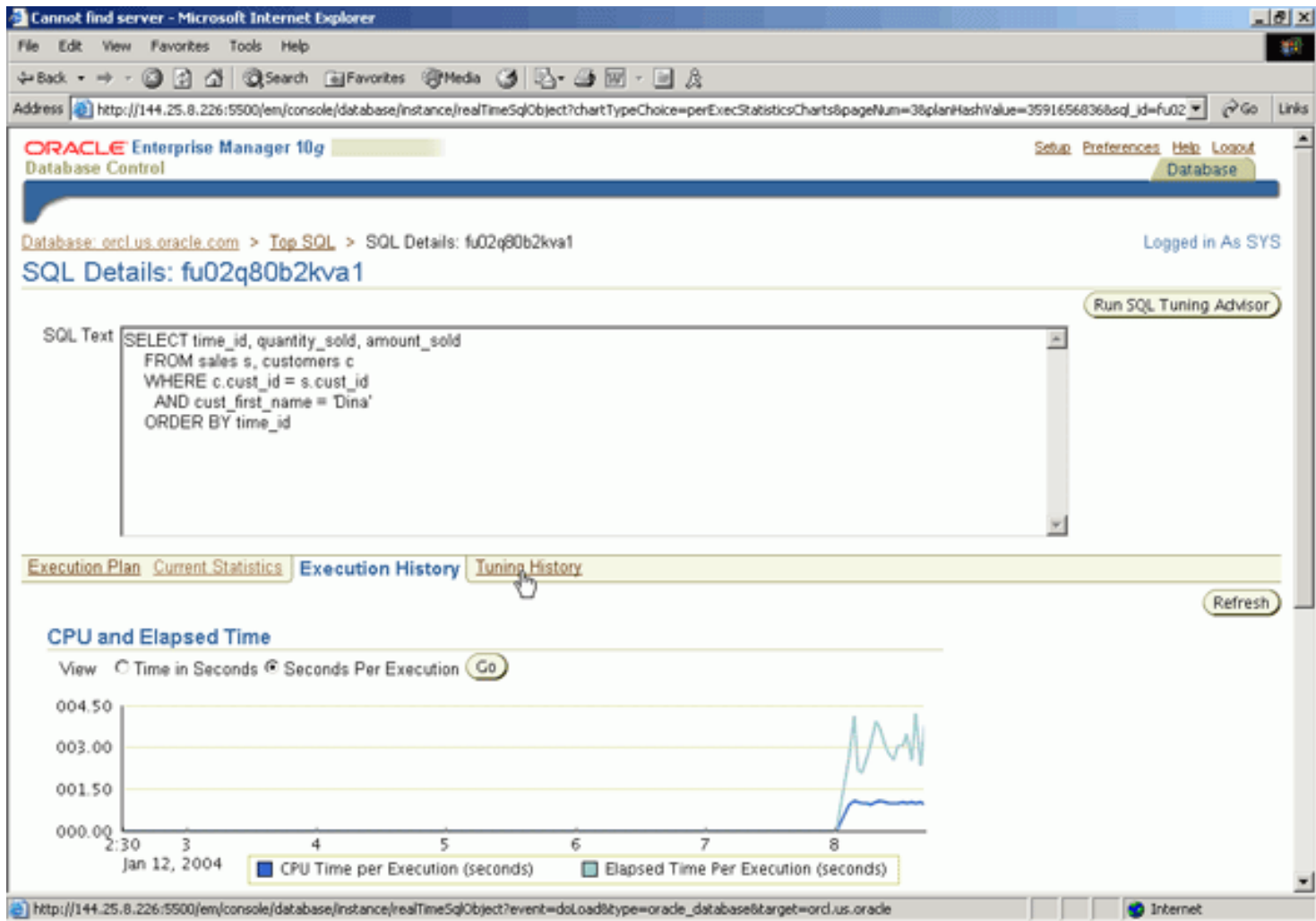
7. The previous tuning recommendations for this SQL statement is displayed. At this time, there are none. You are now ready to tuning the SQL statement using the SQL Tuning Advisor.

Tune a SQL Statement Using the SQL Tuning Advisor

[Back to List](#)

As determined in the previous section on reactive tuning, the targeted SQL statement needs tuning. The SQL Tuning Advisor will tune the execution plan for you. Perform the following:

1. Click **Run SQL Tuning Advisor** .

2. At the Schedule Advisor window, make sure the Scope **Comprehensive** is selected and the job will be scheduled **Immediately** . Click **OK** .

3. The SQL Tuning Advisor will create a task to analyze the SQL statement, and upon completion of this task, displays a set of tuning recommendations.
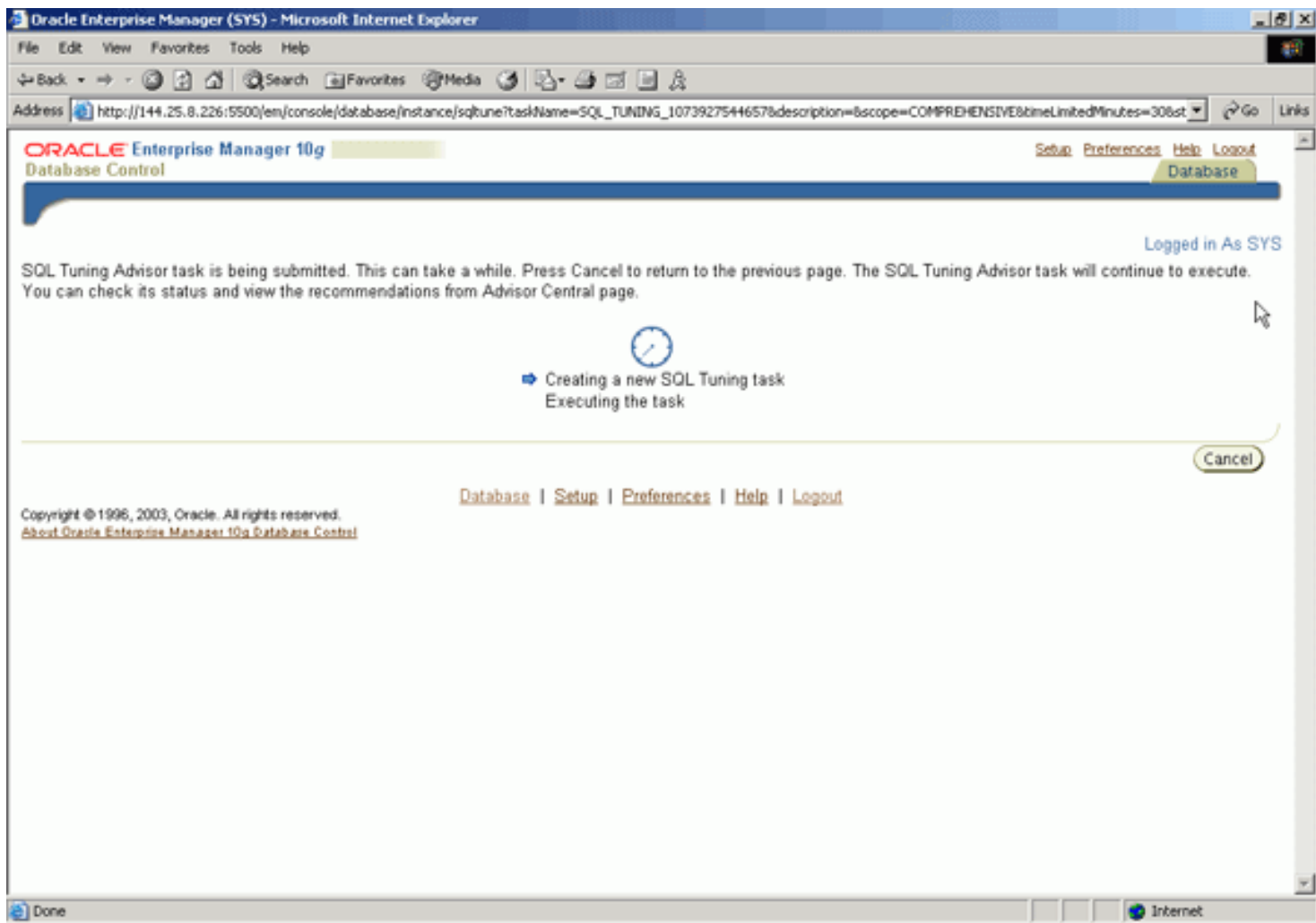
4. Click **View Recommendations** .

5. Click on the **New Explain Plan** button to view the suggested change.

6. As can be seen, the new explain plan removes the full table scans. Click on the browser back button to return to the previous page.

7. Click on the **Implement** button to implement the tuning recommendation.

8. A confirmation page appears indicating that the SQL Profile was successfully created. Click the **Database** breadcrumb.

ORACLE

Enterprise Manager

Database

Database: orcl > Advisor Central > SQL Tuning Results: TASK_00048

Confirmation

The recommended SQL Profile was created successfully.

SQL Tuning Results

Task name	**TASK_00048**
Tuning mode	**COMPREHENSIVE**
SQL ID	**8c2xqj2bhkj7j**
Started at	**Sep 19, 2003 5:41:09 AM**

Task status	**COMPLETED**
Time limit	**1800**
Running time	**6 seconds**
Completed at	**Sep 19, 2003 5:41:15 AM**

Overview of recommendations

(View Recommendations)

⊝ Previous 1-1 of 1 Next ⊝

Select	Parsing Schema	SQL Text	Statistics	SQL Profile	Index	Restructure SQL	Misc	Error
⊙		select time_id, QUANTITY_SOLD, AMOUNT_SOLD from sales s, customers c ...		✔				

Database | Setup | Preferences | Help | Logout

Reviewing SQL Execution Details for a SQL Statement

[Back to List](#)

Now that you have implemented the tuning suggestion, review the SQL statement and its execution details. Perform the following steps.

1. Click on the **Performance** tab.
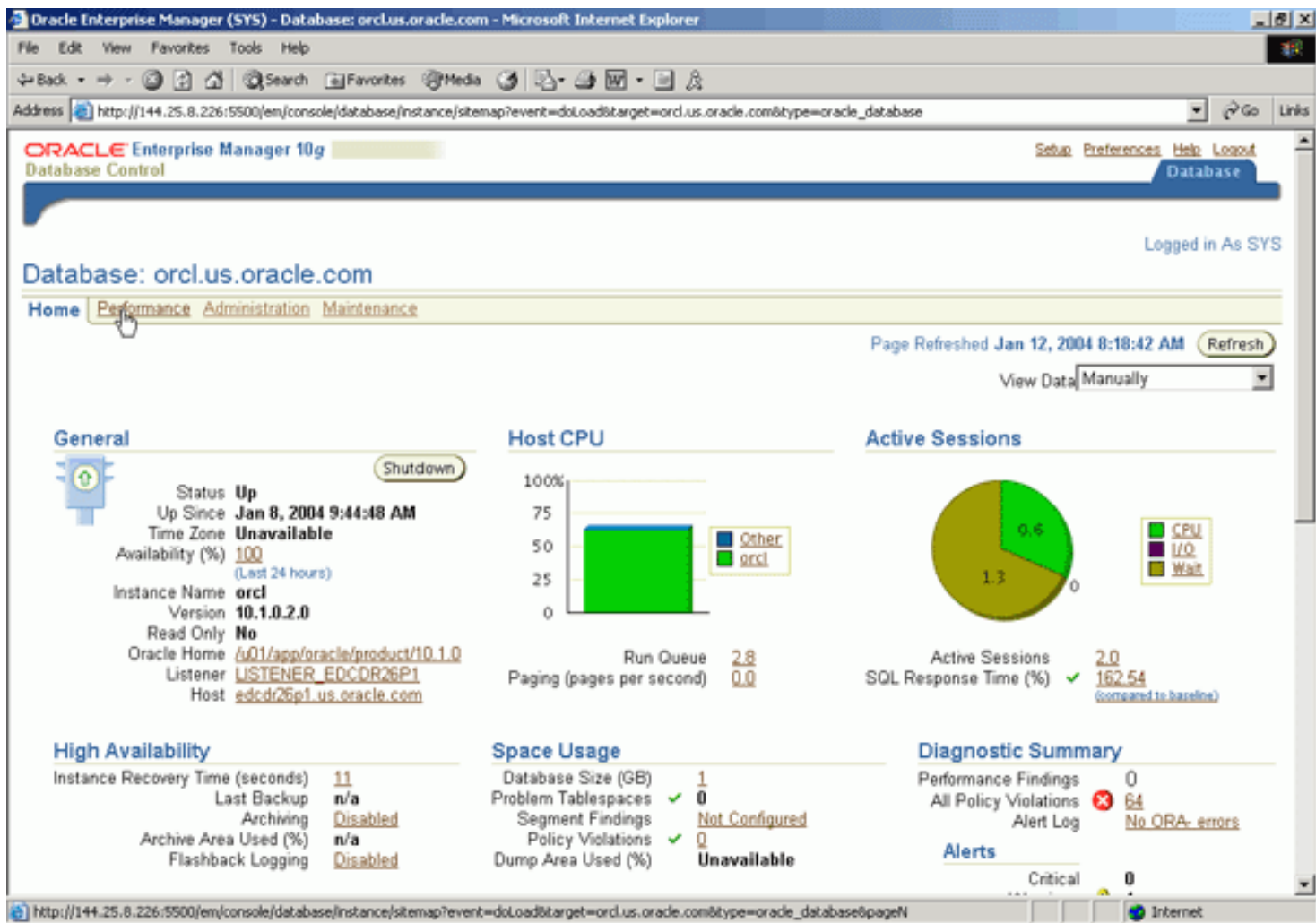
2. Scroll down to the **Sessions: Waiting and Working** chart. Wait for about one minute and observe how the User I/O is decreasing.
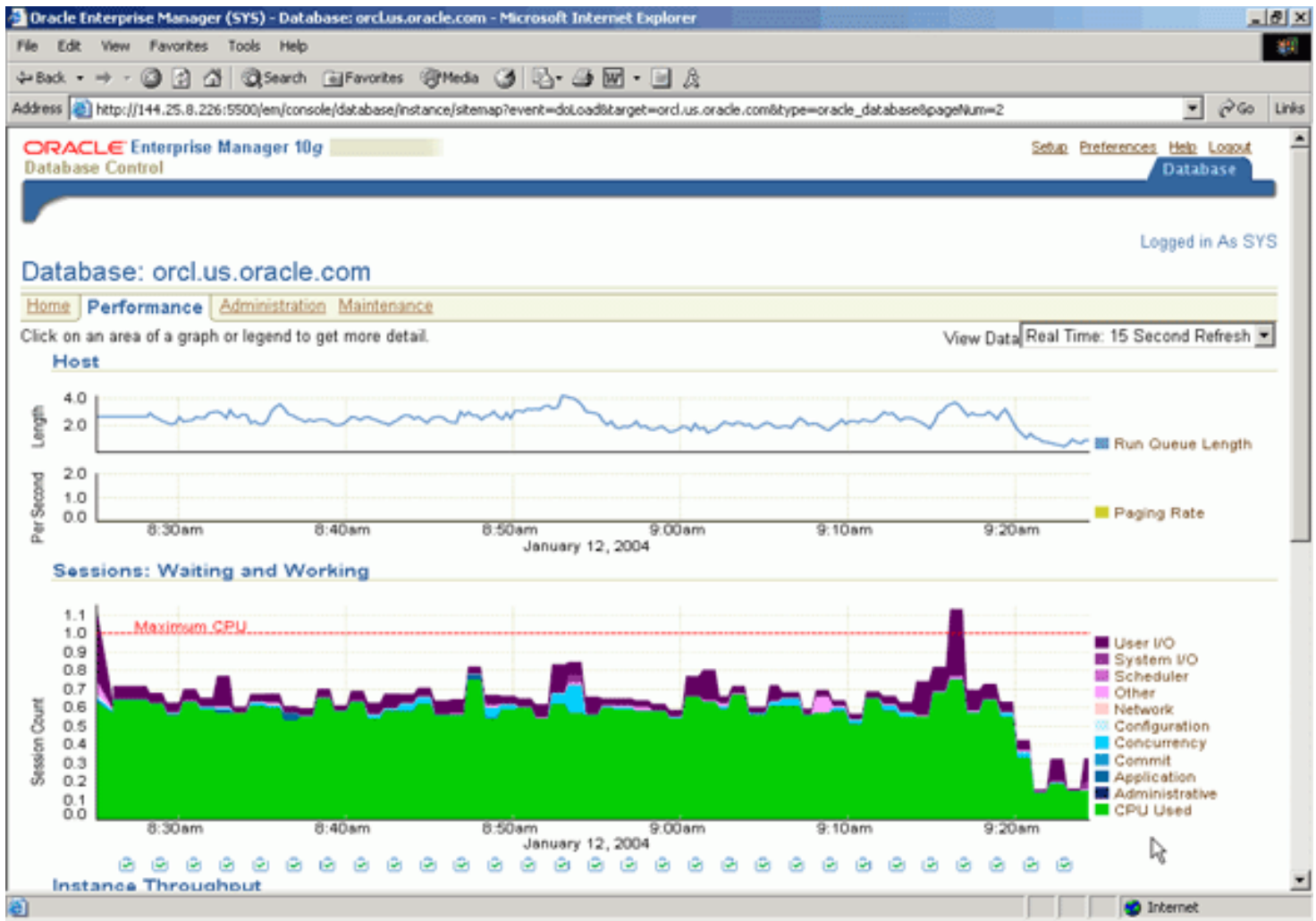
Place the cursor on this icon to hide all screenshots.