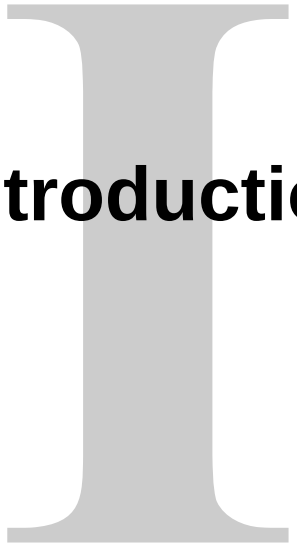


Introduction

Lesson Objectives

After completing this lesson, you should be able to do the following:

- Discuss the goals of the course
- Identify the modular components of PL/SQL:
  - Anonymous blocks
  - Procedures and functions
  - Packages
- Discuss the PL/SQL execution environment
- Describe the database schema and tables that are used in the course
- List the PL/SQL development environments that are available in the course

Course Objectives

After completing this course, you should be able to do the following:

- Create, execute, and maintain:
  - Procedures and functions with `OUT` parameters
  - Package constructs
  - Database triggers
- Manage PL/SQL subprograms and triggers
- Use a subset of Oracle-supplied packages to:
  - Generate screen, file, and Web output
  - Schedule PL/SQL jobs to run independently
- Build and execute dynamic SQL statements
- Manipulate large objects (LOBs)

Course Agenda

Lessons for day 1:

I. Introduction

1. Creating Stored Procedures
2. Creating Stored Functions
3. Creating Packages
4. Using More Package Concepts

Course Agenda

Lessons for day 2:

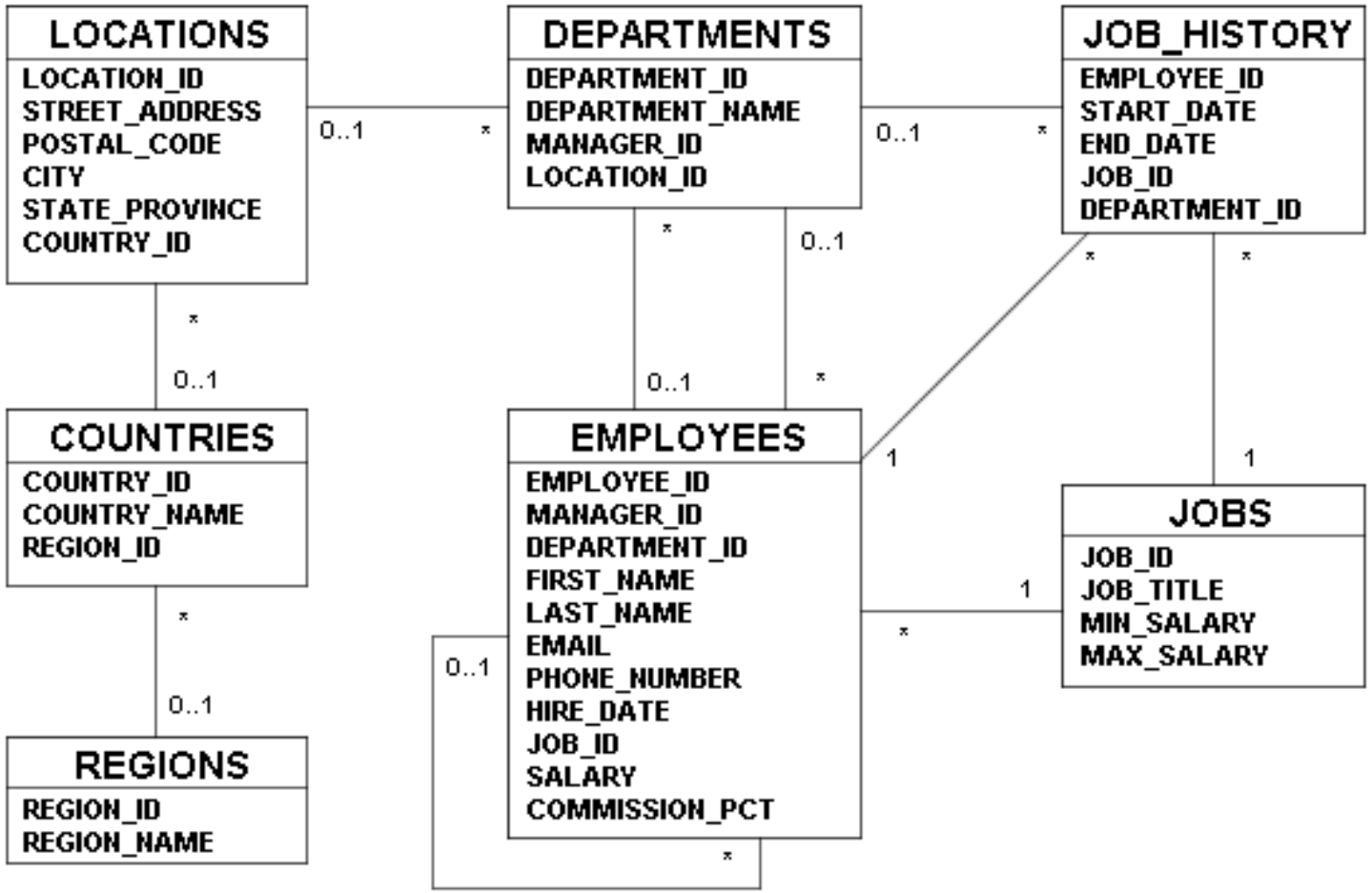
5. Using Oracle-Supplied Packages in Application Development
6. Dynamic SQL and Metadata
7. Design Considerations for PL/SQL Code
8. Managing Dependencies

Course Agenda

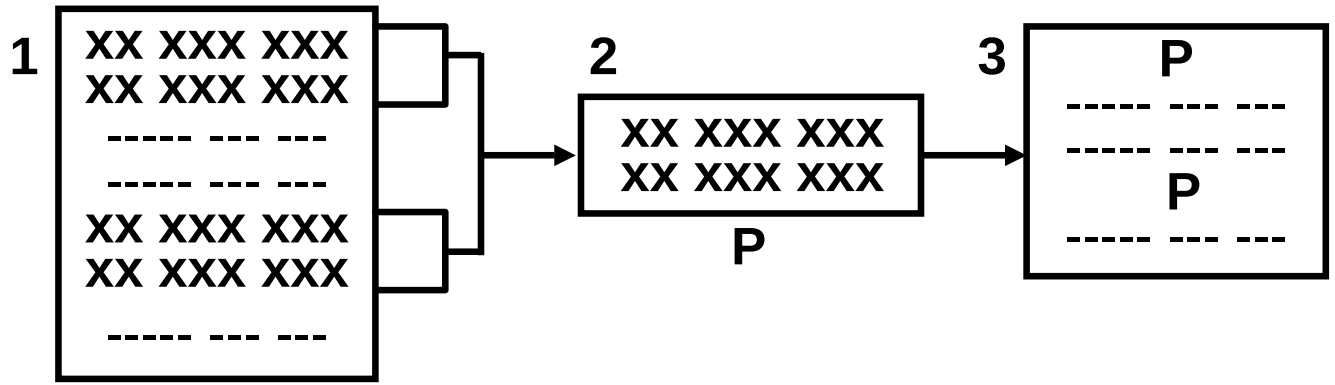
Lessons for day 3:

9. Manipulating Large Objects
10. Creating Triggers
11. Applications for Triggers
12. Understanding and Influencing the PL/SQL Compiler

Human Resources (HR) Schema



Creating a Modularized and Layered Subprogram Design



- Modularize code into subprograms.
  1. Locate code sequences repeated more than once.
  2. Create subprogram P containing the repeated code.
  3. Modify original code to invoke the new subprogram.
- Create subprogram layers for your application.
  - Data access subprogram layer with SQL logic
  - Business logic subprogram layer, which may or may not use data access layer

Modularizing Development with PL/SQL Blocks

- PL/SQL is a block-structured language. The PL/SQL code block helps modularize code by using:
  - Anonymous blocks
  - Procedures and functions
  - Packages
  - Database triggers
- The benefits of using modular program constructs are:
  - Easy maintenance
  - Improved data security and integrity
  - Improved performance
  - Improved code clarity

Review of Anonymous Blocks

Anonymous blocks:

- Form the basic PL/SQL block structure
- Initiate PL/SQL processing tasks from applications
- Can be nested within the executable section of any PL/SQL block

```
[DECLARE      -- Declaration Section (Optional)
  variable declarations; ... ]
BEGIN         -- Executable Section (Mandatory)
  SQL or PL/SQL statements;
[EXCEPTION    -- Exception Section (Optional)
  WHEN exception THEN statements; ]
END;          -- End of Block (Mandatory)
```

Introduction to PL/SQL Procedures

Procedures are named PL/SQL blocks that perform a sequence of actions.

```
CREATE PROCEDURE getemp IS  -- header
  emp_id  employees.employee_id%type;
  lname   employees.last_name%type;
BEGIN
  emp_id := 100;
  SELECT last_name INTO lname
  FROM EMPLOYEES
  WHERE employee_id = emp_id;
  DBMS_OUTPUT.PUT_LINE('Last name: '||lname);
END;
/
```

Introduction to PL/SQL Functions

Functions are named PL/SQL blocks that perform a sequence of actions and return a value. A function can be invoked from:

- Any PL/SQL block
- A SQL statement (subject to some restrictions)

```
CREATE FUNCTION avg_salary RETURN NUMBER IS
  avg_sal employees.salary%type;
BEGIN
  SELECT AVG(salary) INTO avg_sal
  FROM EMPLOYEES;
  RETURN avg_sal;
END;
/
```

Introduction to PL/SQL Packages

PL/SQL packages have a specification and an optional body. Packages group related subprograms together.

```
CREATE PACKAGE emp_pkg IS
  PROCEDURE getemp;
  FUNCTION avg_salary RETURN NUMBER;
END emp_pkg;
/
CREATE PACKAGE BODY emp_pkg IS
  PROCEDURE getemp IS ...
  BEGIN ... END;

  FUNCTION avg_salary RETURN NUMBER IS ...
  BEGIN ... RETURN avg_sal; END;
END emp_pkg;
/
```

Introduction to PL/SQL Triggers

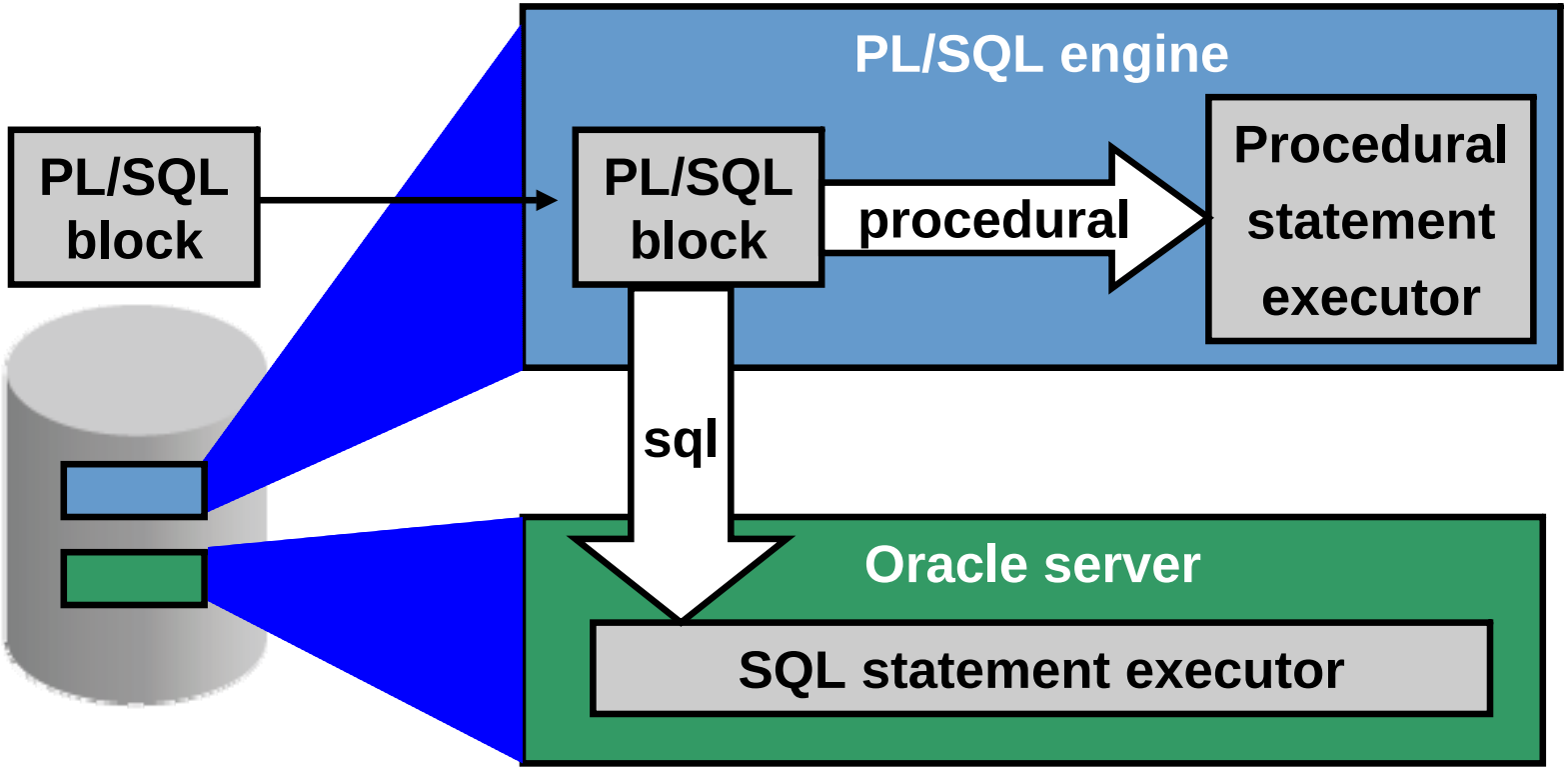
PL/SQL triggers are code blocks that execute when a specified application, database, or table event occurs.

- Oracle Forms application triggers are standard anonymous blocks.
- Oracle database triggers have a specific structure.

```
CREATE TRIGGER check_salary
BEFORE INSERT OR UPDATE ON employees
FOR EACH ROW
DECLARE
  c_min constant number(8,2) := 1000.0;
  c_max constant number(8,2) := 500000.0;
BEGIN
  IF :new.salary > c_max OR
     :new.salary < c_min THEN
    RAISE_APPLICATION_ERROR(-20000,
      'New salary is too small or large');
  END IF;
END;
/
```

PL/SQL Execution Environment

The PL/SQL run-time architecture:



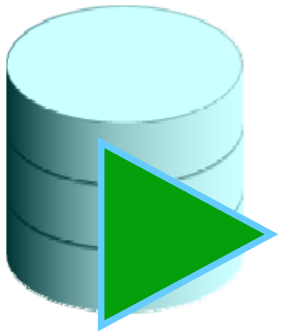
PL/SQL Development Environments

This course provides the following tools for developing PL/SQL code:

- Oracle SQL Developer
- Oracle SQL*Plus (GUI or command-line versions)
- Oracle JDeveloper IDE

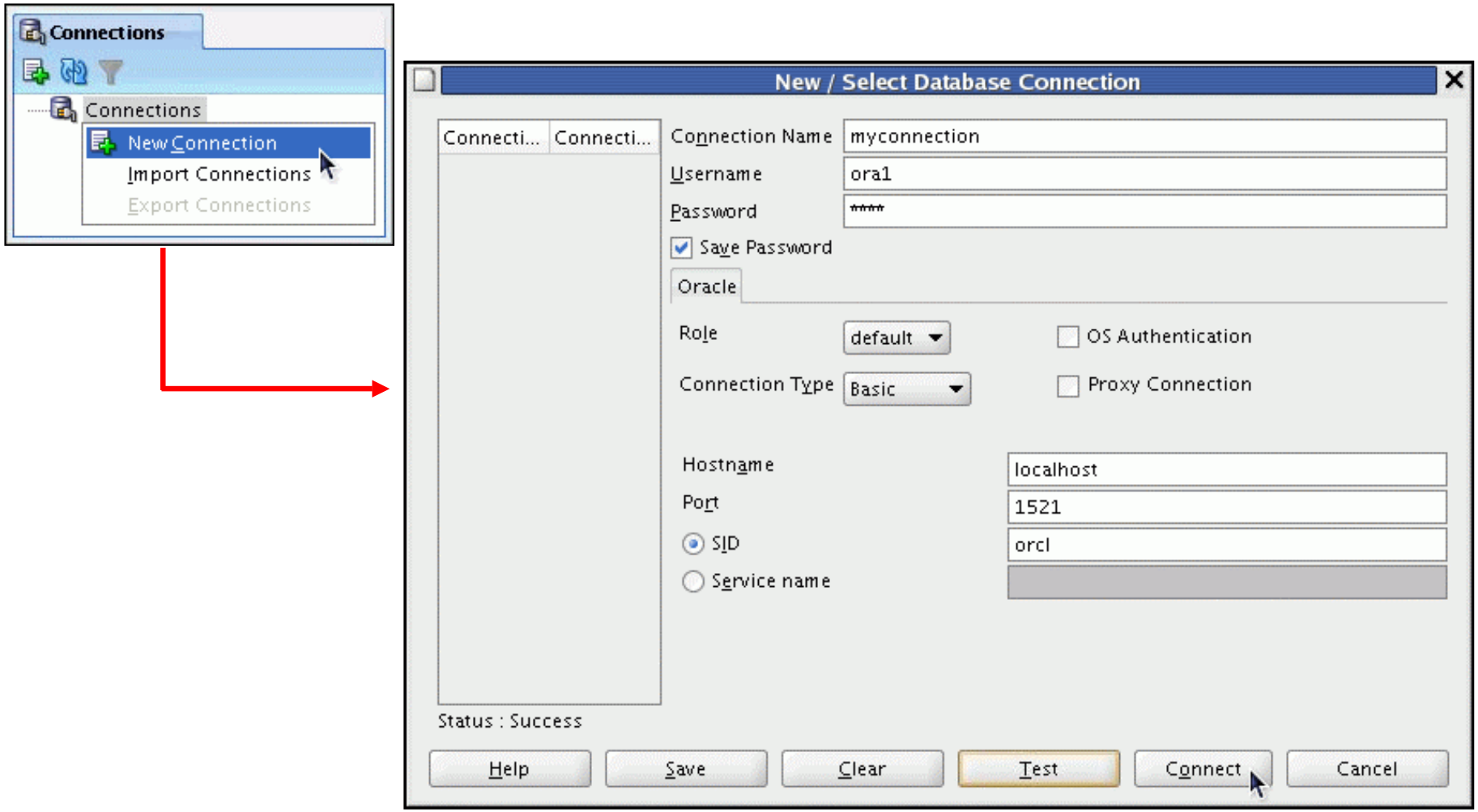
What Is Oracle SQL Developer?

- Oracle SQL Developer is a free graphical tool that enhances productivity and simplifies database development tasks.
- You can connect to any target Oracle Database schema using standard Oracle Database authentication.
- You use SQL Developer in this course.



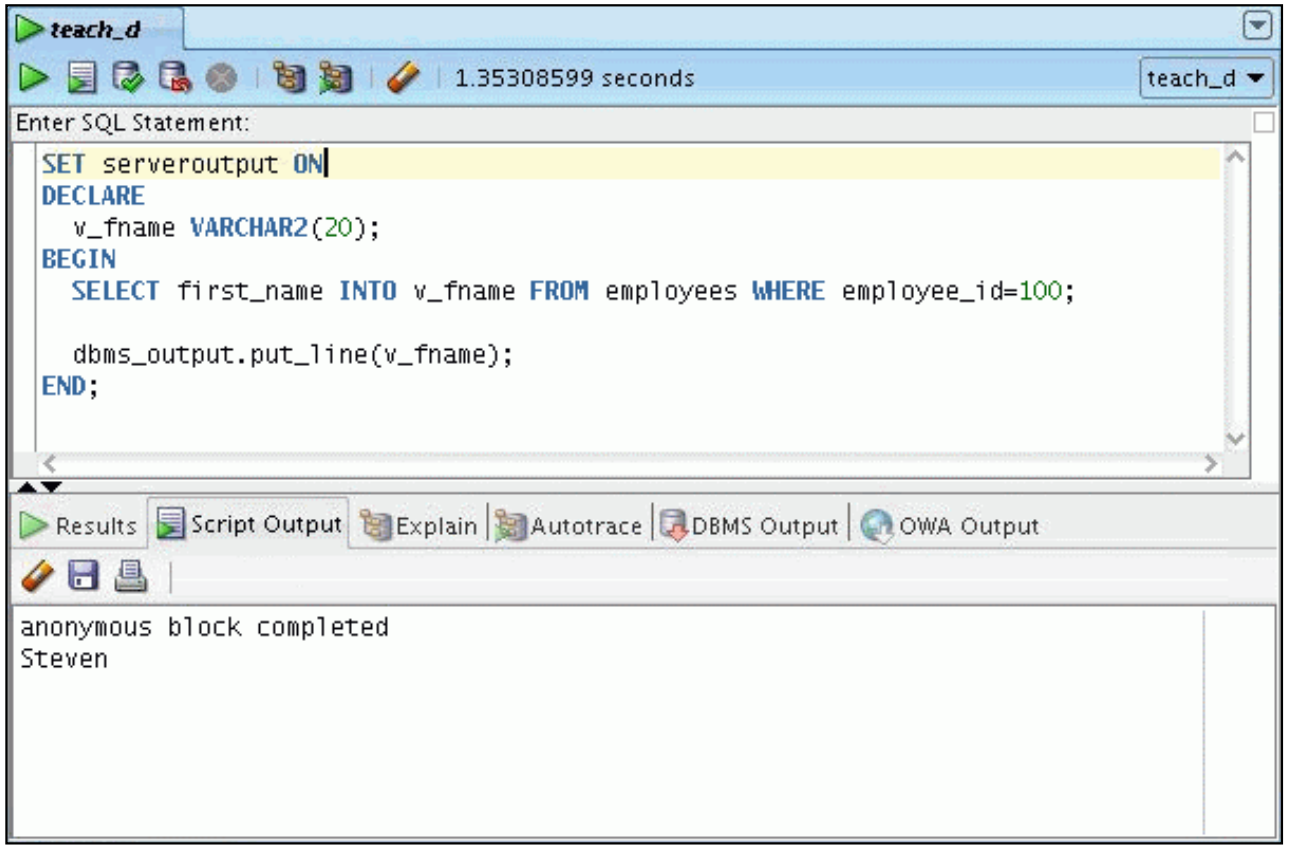
SQL Developer

Creating a Database Connection



Creating an Anonymous Block

Create an anonymous block and display the output of `DBMS_OUTPUT` package statements.



Coding PL/SQL in SQL*Plus

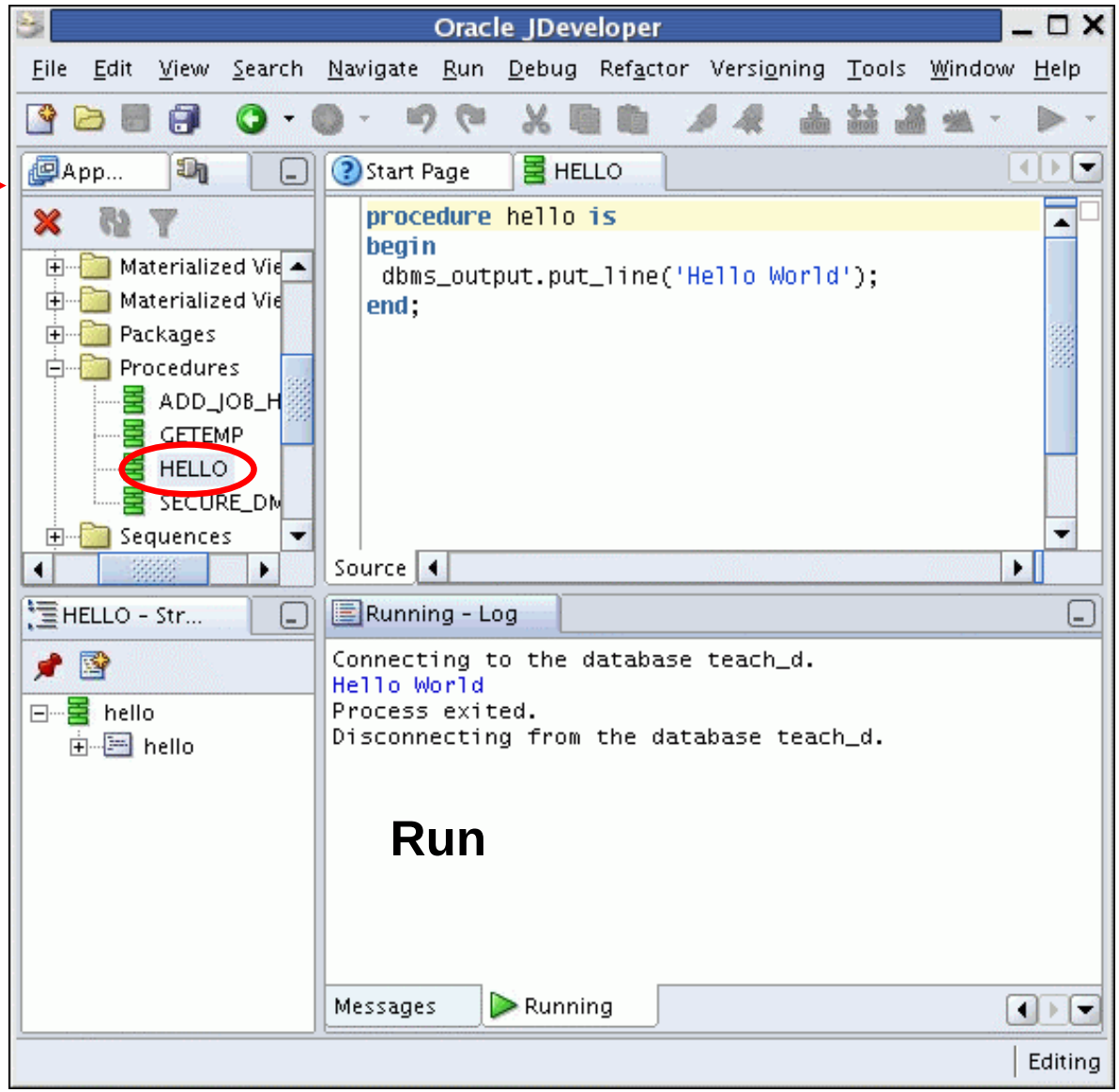
Terminal

File Edit View Terminal Tabs Help

```
SQL*Plus: Release 10.2.0.1.0 - Production on Tue Feb 10 01:27:01 2009

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Enter user-name: teach_d
Enter password:

Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> set serveroutput on
SQL> create procedure hello is
  2  begin
  3   dbms_output.put_line('Hello World');
  4  end;
  5  /

Procedure created.

SQL> execute hello
Hello World

PL/SQL procedure successfully completed.

SQL>
```

Coding PL/SQL in Oracle JDeveloper

JDeveloper



Edit

Summary

In this lesson, you should have learned how to:

- Declare named PL/SQL blocks, including procedures, functions, packages, and triggers
- Use anonymous (unnamed) PL/SQL blocks to invoke stored procedures and functions
- Use SQL Developer or SQL*Plus to develop PL/SQL code
- Explain the PL/SQL execution environment:
  - The client-side PL/SQL engine for executing PL/SQL code in Oracle Forms and Oracle Reports
  - The server-side PL/SQL engine for executing PL/SQL code stored in an Oracle Database

Practice I: Overview

This practice covers the following topics:

- Browsing the HR tables
- Creating a simple PL/SQL procedure
- Creating a simple PL/SQL function
- Using an anonymous block to execute the PL/SQL procedure and function