Introducing Stored Procedures

Objectives

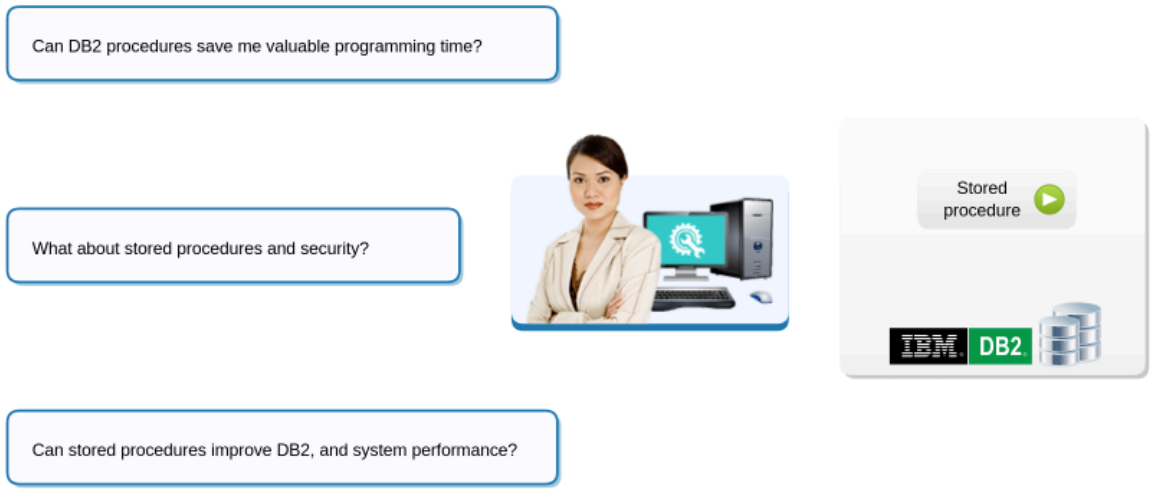
Introducing Stored Procedures

In this module, you will be introduced to Db2 stored procedures, which are an important feature of Db2.

Db2 database administrators (DBAs) and application programmers must understand stored procedures so they can develop, implement, and maintain them. Managers and help desk staff also require this knowledge to manage problems and projects involving stored procedures.

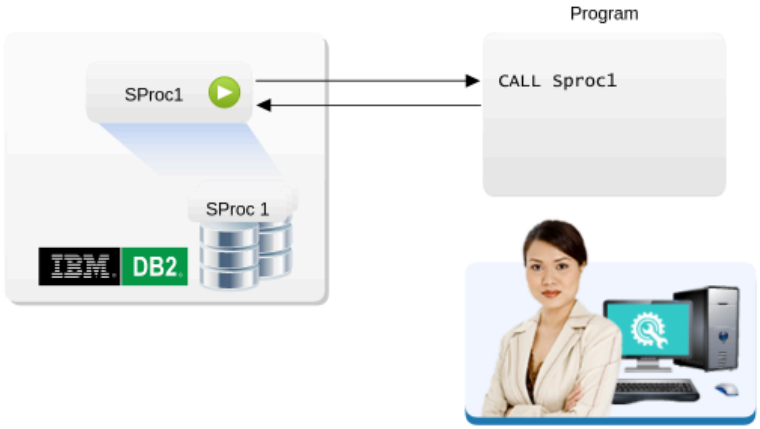
After completing this module, you will be able to identify:

- The Benefits of a Db2 Stored Procedure
- Db2 Stored Procedure Features
- How to Create and Run Db2 Stored Procedures



You are likely to have encountered commonly used Db2 code in your applications. In this course we will look at how this code, which can be based on a programming language such as C, C++, Java, or more commonly just SQL, is saved and invoked as a stored procedure.

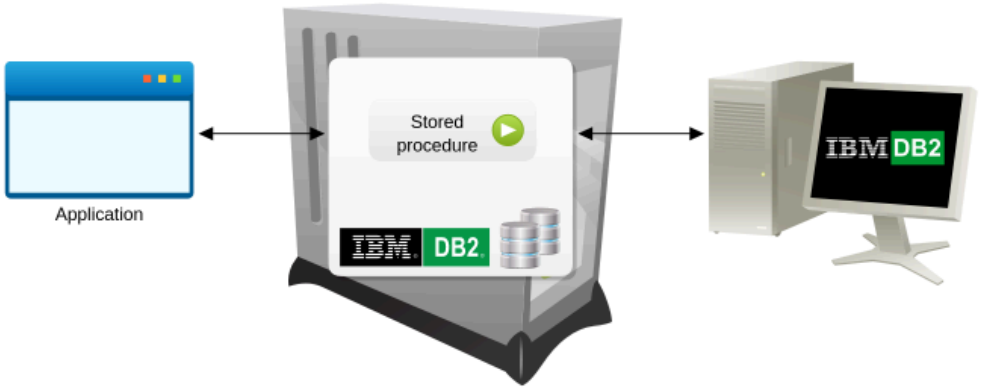
You will also look at the benefits associated with implementing stored procedures.



Stored procedures are called from external programs through standard SQL calls, and are designed to execute host language statements and SQL statements on the local Db2 server.

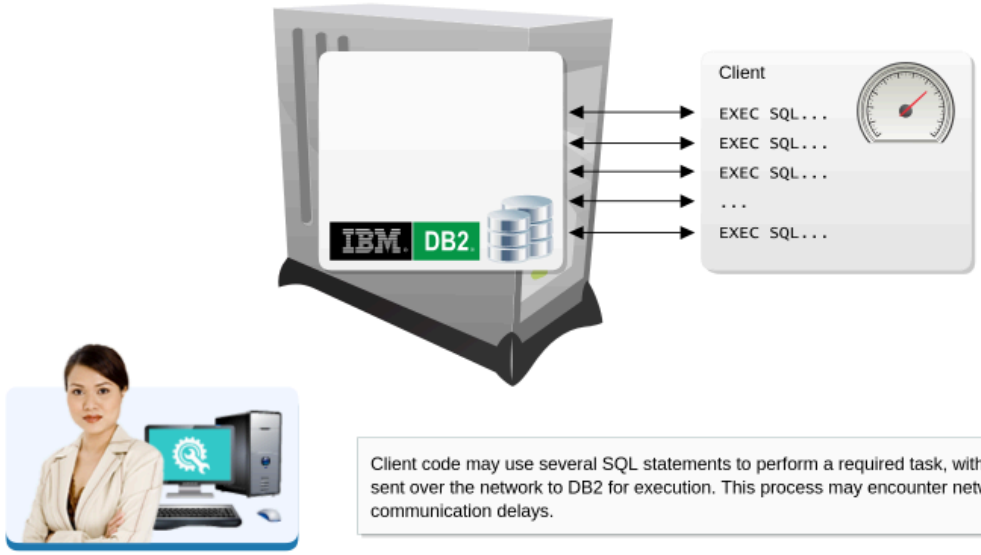
Stored procedures can do anything that a program issuing SQL statements can, and often more. They can be as simple as one or more SQL statements, or they can be compiled programs in languages such as Java, C/C++/C#, assembly, REXX, PL/I, and COBOL. The languages that are supported depend on the platform on which Db2 runs.

Db2 stored procedures are supported by Db2 on all platforms, including z/OS, Microsoft Windows, IBM i, and UNIX.



Like Db2 databases, stored procedures can be accessed from applications on the same system, or other systems that use data-sharing features like Db2 DRDA.

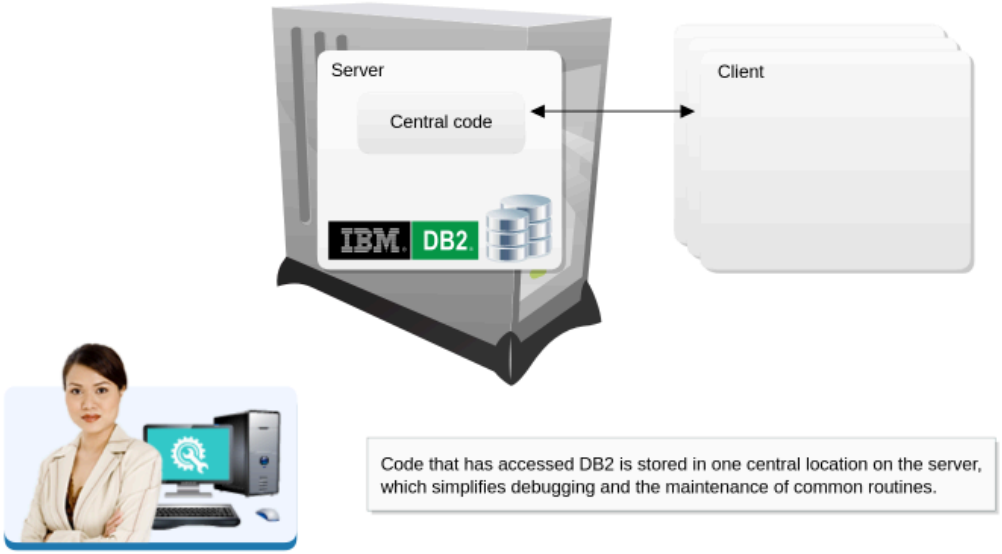
Stored procedures are not unique to Db2. Most other relational database systems, such as Microsoft SQL Server, Oracle, Adabas, and MySQL, support some form of stored procedure.



Client code may use several SQL statements to perform a required task, with each line needing to be sent over the network to DB2 for execution. This process may encounter network traffic resulting in communication delays.

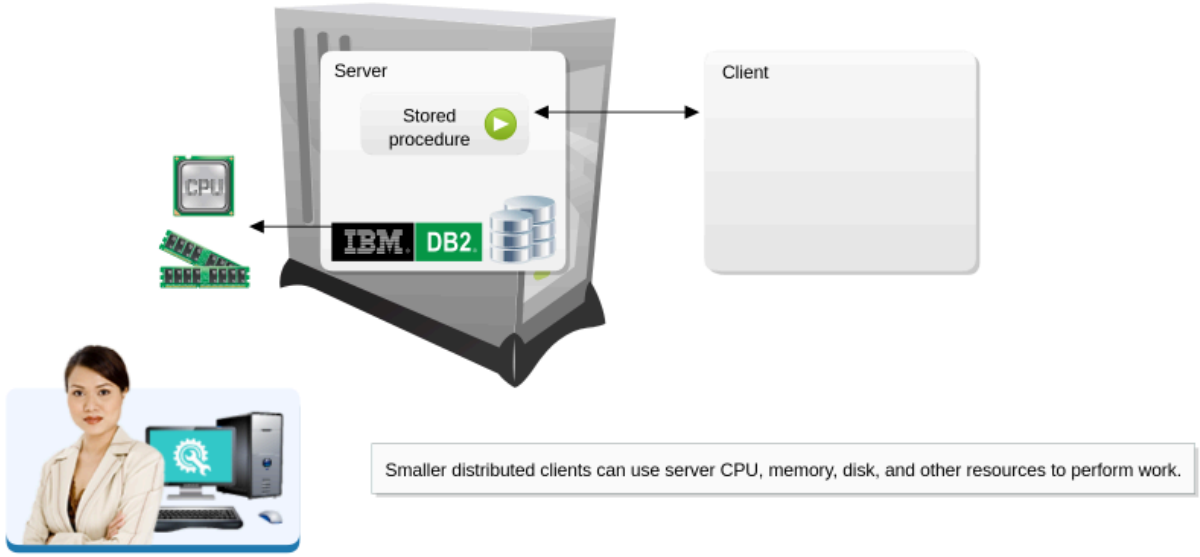
Stored procedures can provide the programmer, and the operating system running Db2 with many benefits

Click Play for some examples of these benefits.



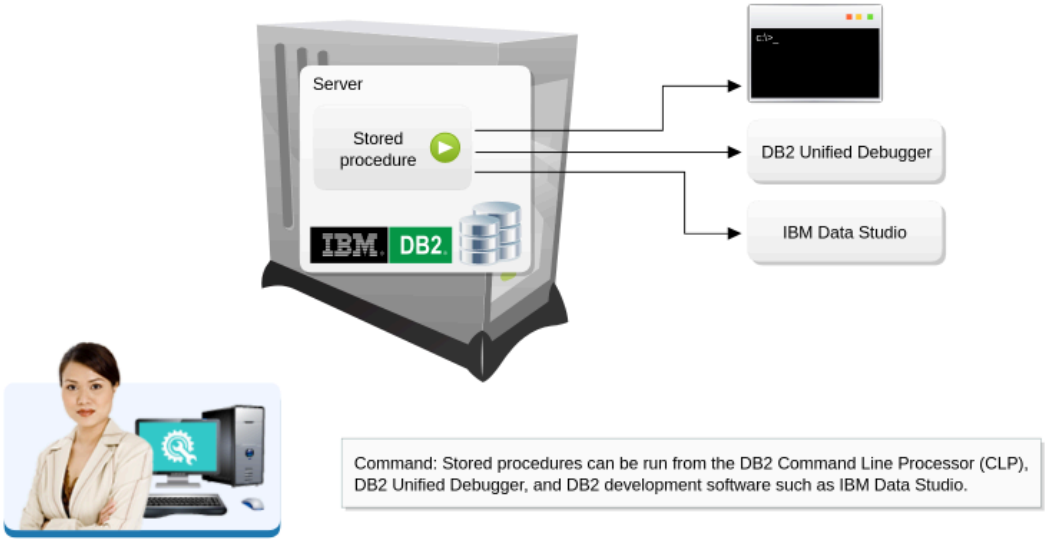
Stored procedures centralize the business logic that has been implemented in other applications, which simplifies administration and maintenance.

Click **Play** to see some advantages of this centralization.



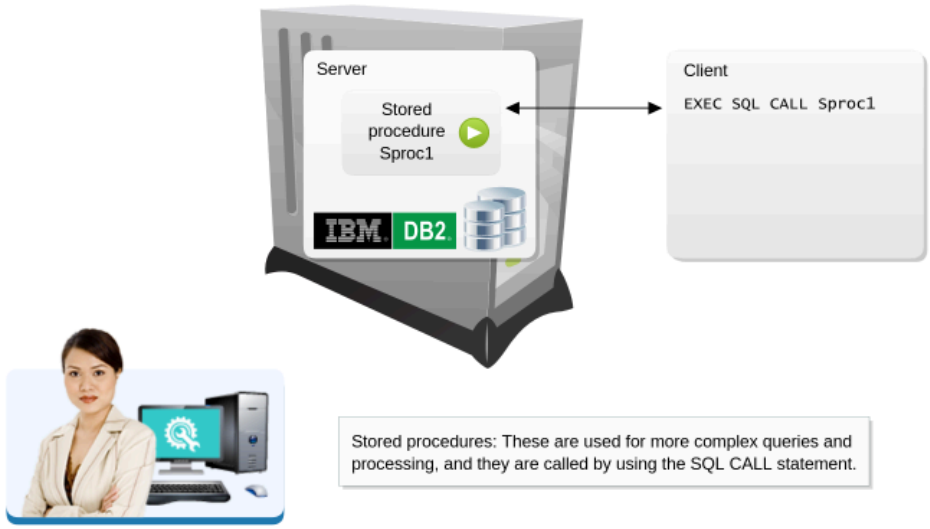
Stored procedures also provide additional functionality for Db2 users.

Click Play to see some of the advantages of these functional enhancements.



Although Db2 stored procedures are usually called from programs, they can also be called from user-defined functions (UDFs), and from other stored procedures and administrator facilities.

Click **Play** to see an example of this concept.



UDFs are program code that can be called as a function in a SQL statement and executed within Db2. They are designed to perform smaller processing tasks. A UDF in Db2 can access Db2 tables, but usually performs small queries.

Stored procedures are designed for larger tasks and more complicated queries, and they are called by using the SQL CALL statement. They often have access to Db2 functions that UDFs do not.

Click Play to see an example of this concept.

Stored procedures can be called by programs from SQL. They can also be called by UDFs, and other stored procedures and administrator facilities.

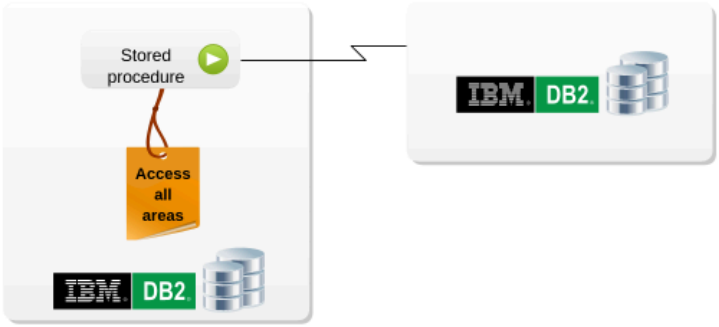
Stored procedures perform larger tasks than user-defined functions (UDFs), and they are called in a different way.

Stored procedures can improve performance, control, and functionality, and can be run from local and remote applications.

DB2 stored procedures are programs that are stored in DB2 databases and run within DB2. They can do everything that a program using SQL statements can, and more.

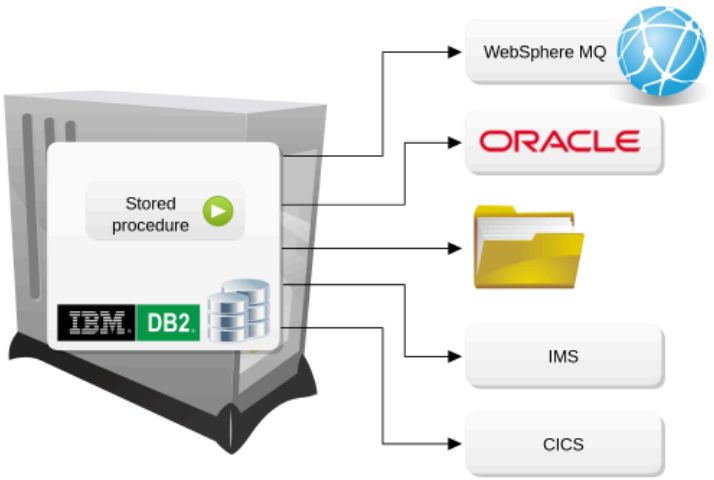
Stored procedures are supported by DB2 on all platforms, and they can be native SQL statements or more sophisticated programs in languages like C, Java, and COBOL.

You have now been introduced to Db2 stored procedures and seen some of the advantages they offer. Next, you will look at the features of Db2 stored procedures.



Db2 stored procedures can do anything that a program running SQL can do, including creating, querying, updating, and deleting tables, and calling stored procedures and UDFs.

If Db2 is sharing databases by using data sharing groups, Distributed Relational Database Architecture (DRDA), or the distributed data facility (DDF), stored procedures can also access remote databases.



Db2 systems also have platform-specific features that allow access to resources such as; other database systems, files, and WebSphere MQ queues. Db2 stored procedures can use these features.

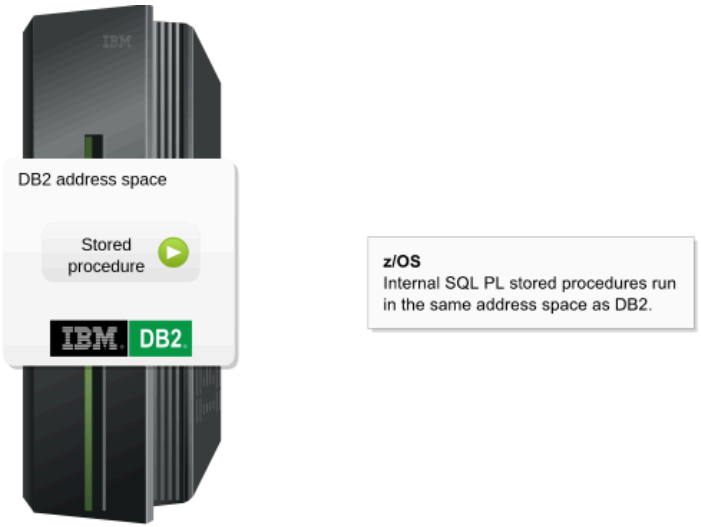
This means that distributed clients can call Db2 stored procedures to access these platform-specific features.

External stored procedure
Usually written in high-level languages (HLL) like C, COBOL, .NET, and Java, and generally slower than internal stored procedures.

The three types of stored procedure are external stored procedures, external SQL procedures and internal SQL procedures. With internal SQL procedures, also known as native SQL procedures, the stored procedure itself is stored in Db2 databases.

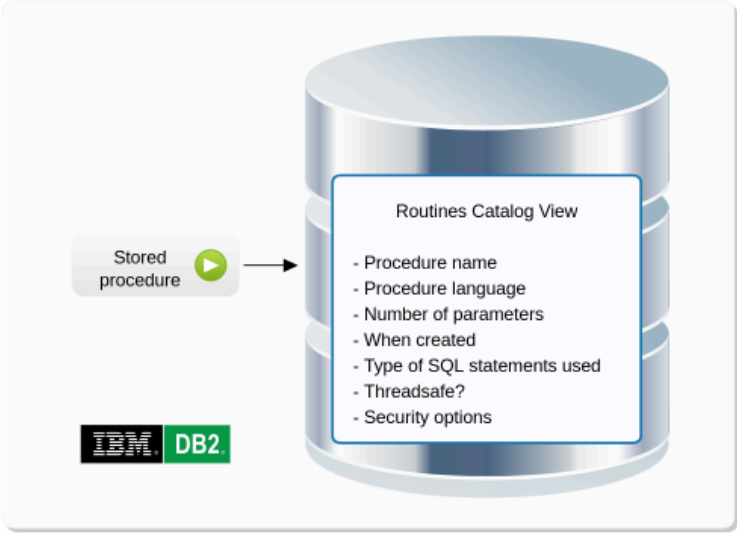
With external stored procedures, the stored procedure source and executable are not stored in a Db2 database but in external files, and the stored procedure is loaded into Db2 when it is run.

Click Play to see an example of this concept.



Stored procedures can run in the same address space as Db2, or in a separate address space. This depends on the platform, the type of stored procedure, and whether the stored procedure has been defined as fenced or non-fenced.

Click Play to see an example of this concept.



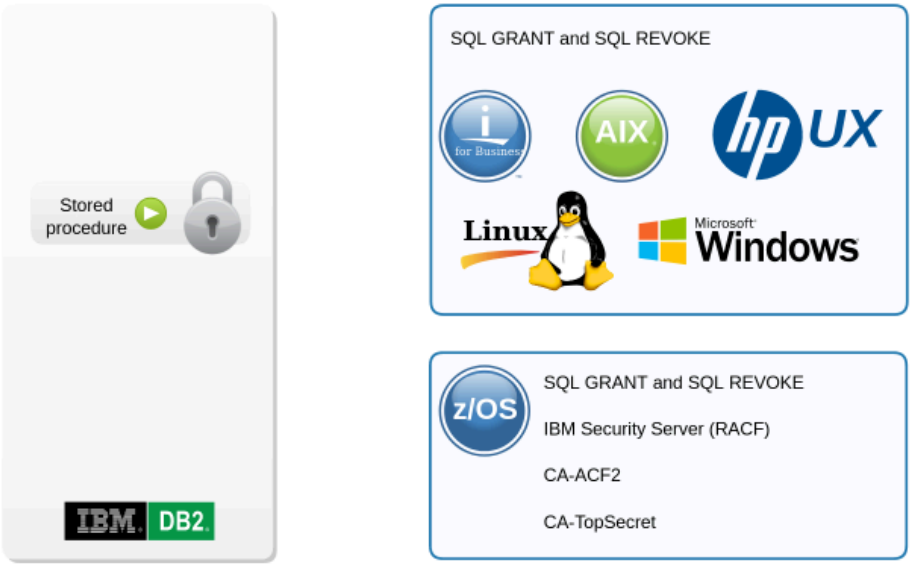
Every stored procedure has an entry in the Routines view of the Db2 catalog: SYSIBM.SYSROUTINES in z/OS and SYSCAT.ROUTINES on Microsoft Windows and UNIX.

Common SQL API procedures

Db2 supplies standard stored procedures for auditing and administrative tasks. Many of these are platform-dependent and cannot be ported to Db2 on other platforms.

Db2 also supplies common SQL API procedures, which are identical for Db2 across IBM platforms and are portable. Standard XML is used for input parameters and output.

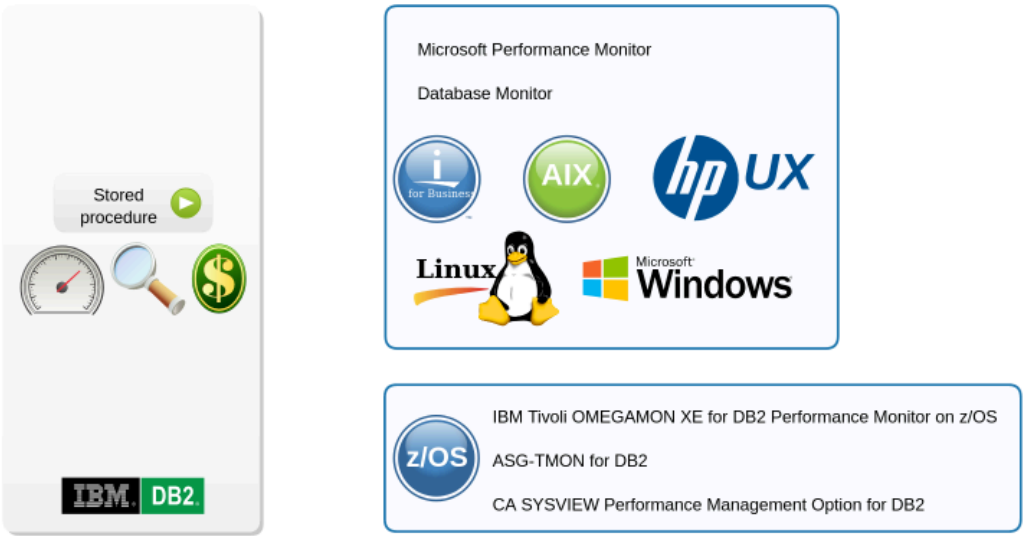
Click Play to see an example of this concept.



Stored procedures are secured in the same way as other Db2 resources:

- On z/OS using SQL GRANT and REVOKE or an External Security Manager like IBM Security Server (RACF) or CA-ACF2.
- On Windows, UNIX and System i using SQL GRANT and REVOKE.

Called procedures run with the authority of the calling program unless their parameters indicate they must run under a different userid.



The management of stored procedures is similar to the management of other Db2 resources. Standard Db2 monitoring functionality, such as the monitoring table functions in Microsoft Windows and UNIX, can also be used with stored procedures. Stored procedure accounting is the same as accounting for Db2 tables.

There is a range of Db2 monitoring products available for the System z platform, some of which are shown above. These can assist in identifying the performance of not just the overall Db2 system, but individual components such as stored procedures also.

Internal stored procedures must be written in SQL PL and are usually faster.

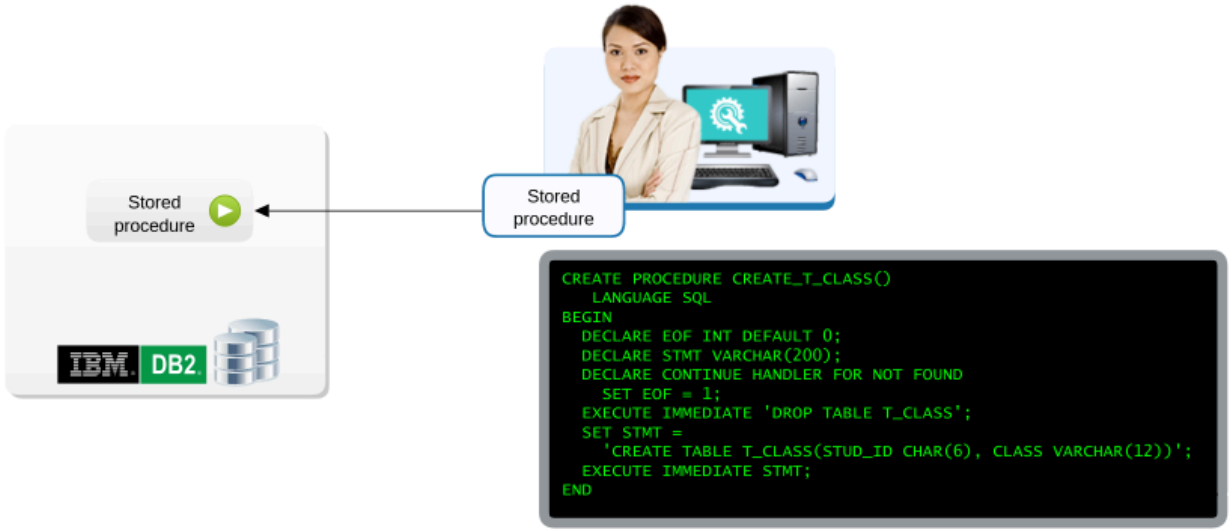
Stored procedure management is very similar to the management of other DB2 resources.

Fenced stored procedures run in their own address space. Non-fenced stored procedures run in the same address space as DB2 and are faster, but they can damage DB2 if errors occur.

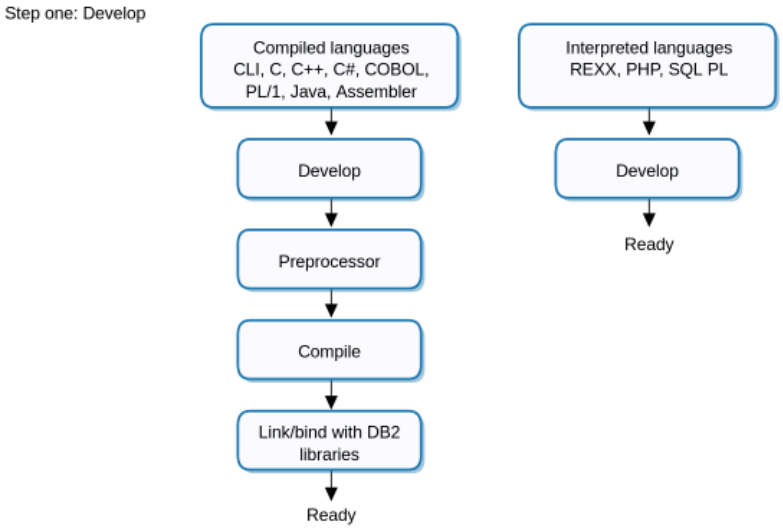
The two types of stored procedures are internal, also known as native because the source is stored in DB2 tables, and external.

In this section, you have explored the different stored procedure types, where they execute, and their security and management.

In the next section, you will learn how to create and run stored procedures.



Although Db2 stored procedures can be sophisticated, creating and running them is easy. You will now see how to create and run a Db2 stored procedure.



The first step in creating a stored procedure is to develop it.

Stored procedures written in high level languages (HLLs) like C and COBOL must be written, compiled and linked as with any Db2 program. On some platforms, a preprocessor must also be used.

Stored procedures written in interpreted languages like PHP have no preprocessing, compile or link steps.

```
CREATE PROCEDURE SP_EXAMPLE (
  IN name CHAR(8),
  OUT address CHAR(128) )
DYNAMIC RESULT SETS 0
NOT DETERMINISTIC
LANGUAGE C
PARAMETER STYLE SQL
DBINFO
FENCED
READS SQL DATA
PROGRAM TYPE SUB
EXTERNAL NAME 'sp.example'
```

NOT DETERMINISTIC indicates that the procedure depends on some state values that will affect the results.

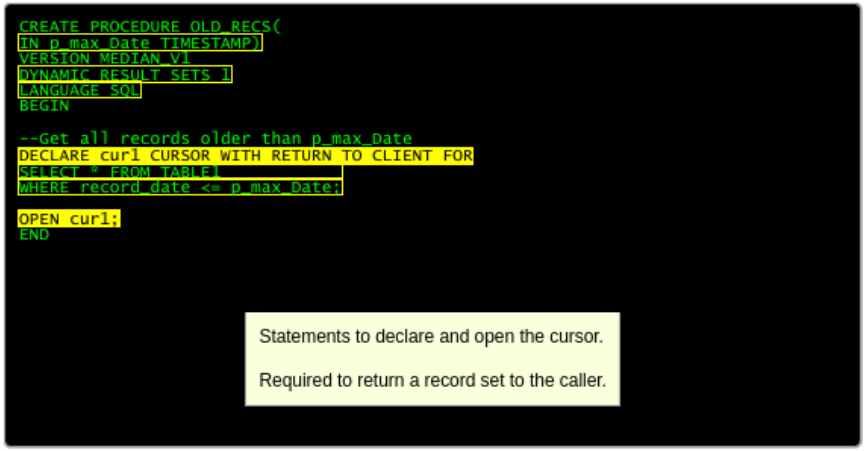
The procedure will be called as if it were a C procedure.

Step one: Develop

Step two: Add to Db2

Once developed, the stored procedure must be added, or defined to Db2 using the SQL create statement. This statement also describes the environment and characteristics of the stored procedure. These values are stored in the Db2 catalog routines view.

Mouse-over the code for more information.



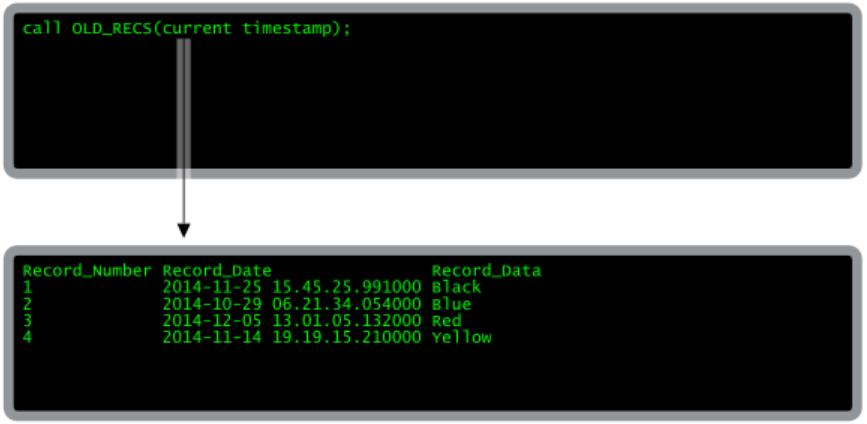
Step one: Develop

Step two: Add to Db2

Internal SQL PL stored procedures are also created with the SQL create statement. This statement also saves the stored procedure's source code in Db2 tables.

An example of a stored procedure that returns all records older than the date and time passed as parameters is shown above.

Mouse-over the code for more information.



Step one: Develop
Step two: Add to DB2
Step three: Call

When the create SQL has been run, the procedure is ready to be called. An example of a call to the OLD_RECS procedure is shown above.

```
DROP PROCEDURE DEC_RTN_RECORDSET;
```

Drop - delete the procedure.

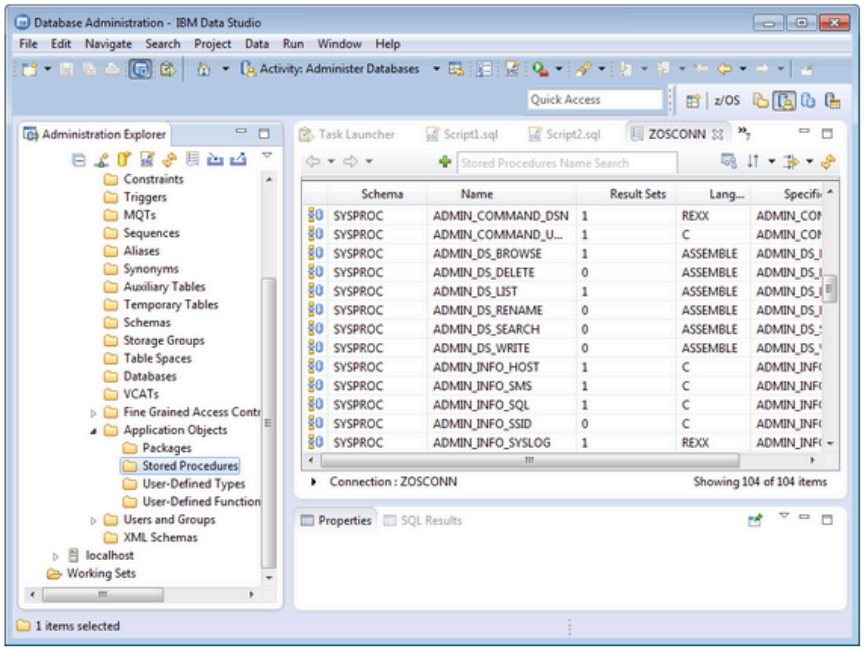
```
ALTER PROCEDURE DEC_RTN_RECORDSET
FENCED;
```

Alter - update some of the procedure properties in the DB2 catalog.

Step 1: Develop
Step 2: Add to DB2
Step 3: Call
Delete/update

Stored procedures can be deleted by using the drop SQL statement. To update or replace a procedure, it must be dropped and then created.

The SQL alter command can be used to alter some of the procedure's properties.



There are many software packages, such as IBM Data Studio, which can be used to develop, test, and debug stored procedures in any supported language.

There are many software products that make developing, adding, and debugging stored procedures easier.

Stored procedures in interpreted languages do not have to be preprocessed, compiled, or linked.

Stored procedures can return parameters, which are single values, and record sets, which are rows and columns.

The create SQL statement adds a stored procedure to DB2, and the environment and characteristics of the stored procedure are specified here.

The drop SQL statement removes a stored procedure from DB2. To update a stored procedure, it must be dropped and then recreated.

Before they can be added to DB2, stored procedures in compiled languages must be compiled and linked like any other program using DB2.

You have now discovered how stored procedures are created, run, and modified.