



# Testing DB2 applications without a DB2 subsystem

Andrew Jepeal



# Increased Testing Pressure

- Agile – Testing is done at end of sprint. Every two weeks not every 3 months
- Test Driven Development – Tests are created first and run more frequently, sometimes daily
- Shifting Testing Left – do it earlier, do it more often

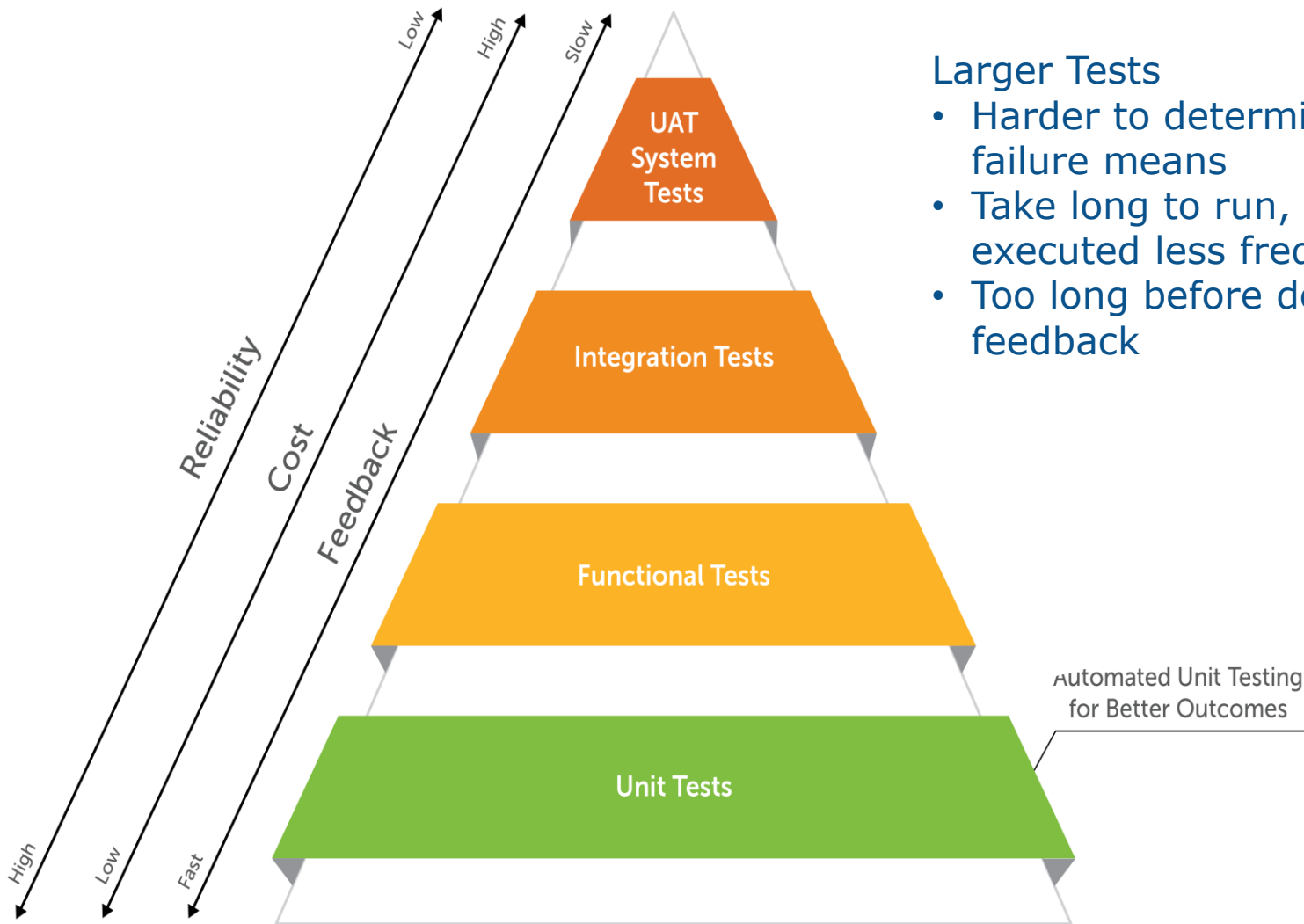
# Increased Testing Challenges

- There's more and they are run more often
  - At the end of every sprint
  - Every day
- Data (tables, databases, datasets) have to be set up in each LPAR
- Data must be refreshed before each test cycle
- New data has to be added to test new functionality
- Barely manageable before the increase in testing

# Increased Testing Solutions?

- Add more staff?
- Reduce testing?
- Automate data refresh?
- Use a Unit test framework that allows:
  - Most of the early testing to be done without needing the DB2 subsystem
  - Testing without needing to manually refresh data
  - Developers to change data to test new functionality.

# Testing



## Larger Tests

- Harder to determine what a failure means
- Take long to run, therefore executed less frequently
- Too long before developer gets feedback

# UAT and System Testing

- What – User Acceptance / Performance / Load
- When – Right before release
- Who – Business Unit, Performance team
- Where – UAT LPAR
- Size – Production copy or production size data
- Why – Gets Business unit acceptance, tests application for performance problems.

# Integration Testing

- What – Whole application or multiple applications
- When – Before release boundary
- Who – Testing Group or QA
- Where – Test LPARs or UAT LPAR
- Size – Usually a subset of production
- Why – Tests the whole application and how the components integrate with each other and possibly other applications.

# Functional Testing

- What – Business function comprised of multiple modules
- When – When all modules are finished
- Who – Developers or Testing Group
- Where – Test LPARs
- Size – Usually a subset of production
- Why – Tests a complete business function but must wait for all changes to be completed. Tests environment also.



# Unit Testing

- What – Individual Unit of code (a module)
- When – Right after code development
- Who – Usually Developers
- Where – Development LPARs and at promotion
- Size – Small number of records – just enough to test all code paths
- Why – Small tests used to check the code logic / algorithms.

# Regression vs New Function

- Not a phase – Done at each phase of testing
- Regression – Rerun tests that have been run before. Make sure working code has not been broken.
- New Function – New tests that make sure the new functions added are working properly.

# A Unit test framework that captures SQL calls and creates stubs.

The screenshot displays the Topaz Workbench interface. On the left, the Project Explorer shows a tree structure for 'NEDB2UG' containing 'Unit Test', 'Interfaces', 'JCL', 'Output', 'Remapping', 'Scenarios', 'Structures', and 'Stubs'. The 'Stubs' folder is expanded, listing various SQL stub files like 'CWYSDB2X\_CWKTDAT\_01.stub'. The main window shows the 'CWYSDB2X\_Scenario.testscenario' with tabs for 'Input Data', 'Stubs', 'Assertions', and 'Properties'. The 'Stubs' tab is active, displaying a list of available stubs and a 'Selected' list. The 'Selected' list includes 'CWYSDB2X\_CWKTDAT\_01.stub', 'CWYSDB2X\_RPTFILE\_FILE\_01.stub', 'CWYSDB2X\_EMPFILE\_FILE\_01.stub', 'CWYSDB2X\_Select\_1\_SQL\_01.stub', 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub', 'CWYSDB2X\_ROLLBACK\_1\_SQL\_01.stub', 'CWYSDB2X\_Insert\_1\_SQL\_01.stub', 'CWYSDB2X\_Update\_1\_SQL\_01.stub', and 'CWYSDB2X\_DATETIME\_01.stub'. Below the stubs, there are fields for 'Alias:', 'Folder:', and 'Project:'. At the bottom, the Console window shows a search bar and a table with columns: Job, Job ID, Owner, Max RC, Description, Class, Po, DD, Records, Step, and DSID.

Selected Object: Testcase CWYSDB2XCase1

# A framework that captures Selects

The screenshot displays the Topaz Workbench interface. The main window shows a project explorer on the left with a tree view containing folders like 'Functional Test', 'Unit Test', and 'Stubs'. The 'Stubs' folder is expanded, showing several SQL stub files, with 'CWYSDB2X\_Select\_1\_SQL\_01.stub' selected. The main pane displays the 'SQL Statement' for this stub: 'EXEC SQL SELECT COUNT(\*) INTO :NUMBER-OF-EMPLOYEES'. Below the statement, there are tabs for 'Selection Criteria', 'Output', 'Return Code', and 'Properties'. The 'Output' tab is active, showing a table with columns 'SQL Selection Criteria', 'Location', 'Value', and 'NULL I...'. The table contains one row: 'EMP\_NUM' with a value of '00000'. At the bottom, there is a 'Search' panel with a search box and a table of jobs. The table has columns: Job, Job ID, Owner, Max RC, Description, Class, Po, DD, Records, Step, and DSID.

Topaz Workbench

File Edit Navigate Search Project Macro Tools Run Configure Compuware Topaz Connect Window Help

Host Explorer Xpediter Total Test Code Coverage Program Analysis Hiperstation Application Auditing

Host Explorer Project Explorer

NEDB2UG

- Functional Test
- Unit Test
  - Interfaces
  - JCL
  - Output
  - Remapping
  - Scenarios
  - Structures
  - Stubs
    - CWYSDB2X\_CWKTDATDATE\_01.stub
    - CWYSDB2X\_DATETIME\_01.stub
    - CWYSDB2X\_EMPFILE\_FILE\_01.stub
    - CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub
    - CWYSDB2X\_Insert\_1\_SQL\_01.stub
    - CWYSDB2X\_ROLLBACK\_1\_SQL\_01.stub
    - CWYSDB2X\_RPTFILE\_FILE\_01.stub
    - CWYSDB2X\_Select\_1\_SQL\_01.stub
    - CWYSDB2X\_Update\_1\_SQL\_01.stub
  - Suites
- SonarLint cwcc [CWCC]
- SonarLint WPAAWJ [SonarLint]
- SQL Editor [CWCC]
- Topaz Total Test 18.2.5 DB2
- TTT DB2-CW01
- TTT 18.2
- TTT 18.3.1
- TTT Batch CWXTCOB CWC2

Selected Object: Data Select

SQL Statement

```
EXEC SQL SELECT COUNT(*)
INTO :NUMBER-OF-EMPLOYEES
```

Selection Criteria Output Return Code Properties

<Quick filter>

SQL Selection Criteria	Location	Value	NULL I...
EMP_NUM			
abc EMP_NUM	1-5	00000	

Console Properties Contents JES Explorer Event History References FTP Client

Search Search to find matching jobs. type filter text Select a job to view the output DDs.

Job	Job ID	Owner	Max RC	Description	Class	Po	DD	Records	Step	DSID

# One that can handle Cursors

The screenshot displays the Topaz Workbench interface. The main window shows a project explorer on the left with a tree view containing folders like 'Functional Test', 'Unit Test', and 'Stubs'. The 'Stubs' folder is expanded, showing several SQL stub files, with 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub' selected. The central pane shows the configuration for this SQL Cursor, including a description, SQL statements for opening and fetching data, and a table of selection criteria.

The bottom pane shows a search interface for jobs. The search criteria are: Prefix: [empty], Owner: [empty]. The search results table is as follows:

Job	Job ID	Owner	Max RC	Description	Class	DD	Records	Step	DSID

The status bar at the bottom indicates 'Selected Object: Data Open'.

# Inserts

Topaz Workbench

File Edit Navigate Search Project Macro Tools Run Configure Compuware Topaz Connect Window Help

Quick Access Host Explorer Xpediter Total Test Code Coverage Program Analysis Hiperstation Application Auditing

Host Explorer Project Explorer CWYSDB2X\_Insert\_1\_SQL\_01.stub

- NEDB2UG
  - Functional Test
  - Unit Test
    - Interfaces
    - JCL
    - Output
    - Remapping
    - Scenarios
    - Structures
    - Stubs
      - CWYSDB2X\_CWKTDAT\_01.stub
      - CWYSDB2X\_DATETIME\_01.stub
      - CWYSDB2X\_EMPFILE\_FILE\_01.stub
      - CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub
      - CWYSDB2X\_Insert\_1\_SQL\_01.stub
      - CWYSDB2X\_ROLLBACK\_1\_SQL\_01.stub
      - CWYSDB2X\_RPTFILE\_FILE\_01.stub
      - CWYSDB2X\_Select\_1\_SQL\_01.stub
      - CWYSDB2X\_Update\_1\_SQL\_01.stub
    - Suites
    - SonarLint cwcc [CWCC]
    - SonarLint WPAAWJ [SonarLint]
    - SQL Editor [CWCC]
    - Topaz Total Test 18.2.5 DB2
    - TTT DB2-CW01
    - TTT 18.2
    - TTT 18.3.1
    - TTT Batch CWXTCOB CWC2
    - TTT Batch-CWXTCOB-CWCC

Description	SQL Statement
1 Insert	EXEC SQL INSERT INTO YX_DEMOTAB (

Structure	Location	Compa...	Expected Value	Label	Failure Message
EMP_NUM					
abc EMP_NUM	1-5	=	02175	Check for EMP_NUM	Check for EMP_NU...
WA_EMP_TYPE					
abc WA_EMP_TYPE	1-1	=	H	Check for WA_EMP...	Check for WA_EMP...
REGION					
123 REGION	1-2	=	2	Check for REGION	Check for REGION f...
WA_EMP_NAME					
abc WA_EMP_NAME	1-15	=	KATIE WILLIAMS	Check for WA_EMP...	Check for WA_EMP...
WA_EMP_STREET					
abc WA_EMP_STREET	1-15	=	123 MAIN ST	Check for WA_EMP...	Check for WA_EMP...
WA_EMP_CITY					
abc WA_EMP_CITY	1-8	=	COLUMBUS	Check for WA_EMP...	Check for WA_EMP...
WA_EMP_STATE					
abc WA FMP STAF	1-2	=	OH	Check for WA FMP ...	Check for WA FMP ...

Console Properties Contents JES Explorer Event History References FTP Client

Search Search to find matching jobs. type filter text Select a job to view the output DDs.

Prefix:	Job	Job ID	Owner	Max RC	Description	Class	P	DD	Records	Step	DSID
	<										

Selected Object: Data Insert

# Updates

The screenshot displays the Topaz Workbench application window. The interface includes a menu bar (File, Edit, Navigate, Search, Project, Macro, Tools, Run, Configure, Compuware, Topaz Connect, Window, Help) and a toolbar with various icons. The main workspace is divided into several panes:

- Project Explorer:** Shows a tree view of the project structure under 'NEDB2UG', including folders like 'Functional Test', 'Unit Test', and 'Suites', and several '.stub' files. The file 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub' is selected.
- SQL Editor:** Contains two SQL statements:
 

```
EXEC SQL DECLARE EMPLOYEE_CURSOR CURSOR FOR
SELECT
```

```
EXEC SQL UPDATE YX_DEMOTAB
SET (WAGES, OVERTIME, COMM)
```
- Table:** A table with columns: Structure, Location, Compa..., Expected Value, Label, and Failure Message. It displays test results for WAGES, OVERTIME, and COMM.
 

Structure	Location	Compa...	Expected Value	Label	Failure Message
WAGES	1-4	=	95.00	Check for WAGES	Check for WAGES fa...
OVERTIME	1-4	=	0	Check for OVERTIME	Check for OVERTIM...
COMM	1-4	=	0	Check for COMM	Check for COMM fa...
- Search Panel:** Located at the bottom, it includes a search bar and a table with columns: Job, Job ID, Owner, Max RC, Description, Class, DD, Records, Step, and DSID.

The status bar at the bottom indicates 'Selected Object: Data Update'.

# Non-zero SQL codes

The screenshot shows the Topaz Workbench interface. On the left, the Project Explorer shows a tree structure under 'NEDB2UG' with 'Unit Test' expanded to 'Stubs'. The file 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub' is selected. The main editor displays the SQL code for this stub:

```
EXEC SQL DECLARE EMPLOYEE_CURSOR CURSOR FOR  
SELECT  
  
EXEC SQL FETCH EMPLOYEE_CURSOR INTO  
:KTDCCL-DEMOTAB:IKT-DEMOTAB.INDSTRUC
```

Below the code, the execution results are shown in a table:

Selection Criteria	Output	Return Code	Properties
Return Code		Value	
SQLSTATE		02000	
SQLCODE		100	

The 'Fetch' button in the toolbar is highlighted. At the bottom, the JES Explorer shows a search for matching jobs with a table of results:

Job	Job ID	Owner	Max RC	Description	Class	DD	Records	Step	DSID

Selected Object: Data Fetch



# Handles other data and program calls

The screenshot displays the Topaz Workbench application window. The interface includes a menu bar (File, Edit, Navigate, Search, Project, Macro, Tools, Run, Configure, Compuware, Topaz Connect, Window, Help) and a toolbar with various icons. The main workspace is divided into several panes:

- Host Explorer / Project Explorer:** Shows a tree view of the project structure. The selected object is `CWYSDB2X_EMPFILE_FILE_01.stub`.
- Object List:** A table listing objects with the following data:

Description
1 EMPFILE
2 EMPFILE
3 EMPFILE
4 EMPFILE
5 EMPFILE
6 EMPFILE
7 EMPFILE
8 EMPFILE
- Input Data(8) Properties:** A detailed view of the selected object's structure, showing a table of input data with columns for Structure, Location, and Value.

Structure	Location	Value
EMPLOYEE_WORK_AREA		
123 WA_EMP_NUM	1-5	3431
abc WA_EMP_TYPE	6-6	H
123 WA_EMP_REGION	7-7	2
abc WA_EMP_NAME	8-22	GAIL LAWRENCE
WA_EMP_ADDRESS		
abc WA_EMP_STREET	23-37	123 NORTH AVE
abc WA_EMP_CITY	38-45	PLANO
abc WA_EMP_STATE	46-47	TX
abc WA_EMP_ZIP	48-56	57010
WA_HOURLY_EMPLOYEE_DATA		
123 WA_EMP_HOURS	57-58	38
123 WA_EMP_RATE	59-61	25.00
abc FILLERO	62-69	
WA_SALES_EMPLOYEE_DATA		
123 WA_SALES_SALARY	57-60	
- Console:** Displays the message "No consoles to display at this time."

At the bottom of the window, the status bar indicates "Selected Object: Data EMPFILE".

# Quickly lets you see if the test passes or fails

The screenshot displays the Topaz Workbench interface. The main window shows a 'Test Suite Summary' for 'Test Project NEDB2UG'. The summary table indicates 225 tests, 0 failures, 0 errors, and a 100% success rate. Below this, a 'Test Scenarios' table shows the same metrics for the 'CWYSDB2X\_Scenario'.

**Test Suite Summary**

Name	Tests	Failures	Errors	Success rate	Time stamp	Job ID/Output	System
CWYSDB2X_Scenario	225	0	0	100%	2019-09-24 08:40:40	WPAAWJ01(J0786240) Output (Unit Test/Output/History/CWYSDB2X_Scenario.log)	CWCC LPAR

**Test Scenarios**

*Note: scenario statistics are not computed recursively, they only sum up all of its test case numbers.*

Name	Tests	Failures	Errors	Success rate
CWYSDB2X_Scenario	225	0	0	100%

The interface also shows a 'Host Explorer' on the left with a tree view of project files, and a 'Console' at the bottom with a search for jobs on CWCC. The console search results show two jobs: PAAWJ02 and PAAWJ01, both ending normally with a Max RC of 0.

# Lets Developers change test data easily

The screenshot displays the Topaz Workbench interface. On the left, the Project Explorer shows a tree view of test files, with 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub' selected. The main window is titled 'SQL Cursor' and shows the following SQL statements:

```
EXEC SQL DECLARE EMPLOYEE_CURSOR CURSOR FOR  
SELECT  
  
EXEC SQL FETCH EMPLOYEE_CURSOR INTO  
:KTDCLE-DEMOTAB:IKT-DEMOTAB.INDSTRUC
```

Below the statements, the 'SQL Output Variables' table is visible:

SQL Output Variables	Location	Value	NULL I...
EMP_NUM			
abc: EMP_NUM	1-5	03431	Non-N...
WAGE_TYPE			
abc: WAGE_TYPE	1-1	H	Non-N...
REGION			
123: REGION	1-2	2	Non-N...
FIRST_LAST_NAME			
456: FIRST_LAST_NAME			Non-N...
789: FIRST_LAST_NAME_LEN	1-2	13	

At the bottom, the Console window shows search results for 'CWCC' with two jobs found:

Prefix	Job ID	Owner	Max RC	Description	Class	Pos	DD	Records	Step	DSID
b	PAAWJ02	J0786...	WPAAWJ	0	Ended normally	A				
	PAAWJ01	J0786...	WPAAWJ	0	Ended normally	A				

# Lets Developers modify the data structure

The screenshot displays the Topaz Workbench interface. The main window shows the execution of a SQL cursor. The SQL Statement is:

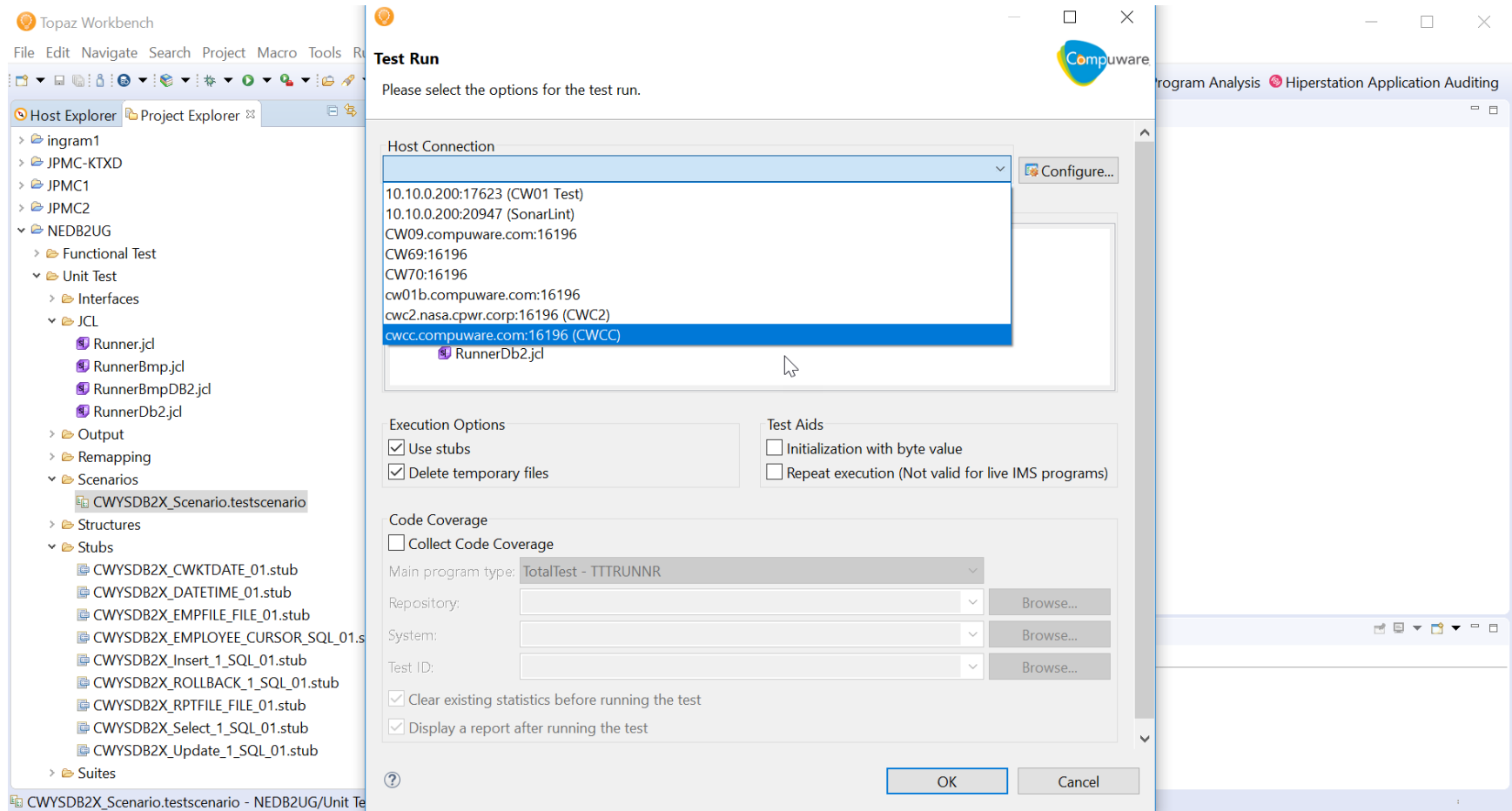
```
EXEC SQL DECLARE EMPLOYEE_CURSOR CURSOR FOR  
SELECT  
  
EXEC SQL FETCH EMPLOYEE_CURSOR INTO  
:KTDCLE-DEMOTAB:IKT-DEMOTAB.INDSTRUC
```

The output is displayed in a table with the following columns: SQL Output Variables, Location, Value, and NULL I... The table contains the following data:

SQL Output Variables	Location	Value	NULL I...
EMP_NUM			
abc EMP_NUM	1-5	03431	Non-N...
WAGE_TYP			
abc WAGE_T		H	Non-N...
REGION			
123 REGION	2		Non-N...
FIRST_LAST_NAME			
FIRST_LAST_NAME			Non-N...
FIRST_LAST_NAME_LEN	1-2	13	

The interface also shows a Project Explorer on the left with a tree view of files, including 'CWYSDB2X\_EMPLOYEE\_CURSOR\_SQL\_01.stub'. A context menu is visible over the 'REGION' row in the output table, with options: 'Expand levels', 'Collapse levels', 'Modify Length...', and 'R Redefine element'. The console at the bottom shows 'No consoles to display at this time.' The status bar at the bottom indicates 'Selected Object: Data Fetch'.

# This creates portable unit tests that can even be run on LPARs without DB2



Let DBAs and Sys Progs focus on their work, not get bogged down in maintaining testing environments.