Test Scripts for the Software Metrics Collection System (SMCS)

Author: Jeffrey Paulson (Team Tortoise)
Date: 11/25/2019

1. DATABASE SCHEMA TESTING

1.1 'FindNullHours' Procedure Unit Test

Purpose of Test: Verify accurate functionality of 'FindNullHours' procedure.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with MySQL Workbench. **Environment Prerequisites:** MySQL Workbench with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS						
STEP	**TEST INPUT**	**EXPECTED RESULTS**	**ACTUAL RESULTS**	**REQUIREMENTS VALIDATED**	**PASS/FAIL**	**INIT**
1.	Run: CALL FindNullHours("BadEmail@not_an_email.com");	Empty table as a result. No errors producing table.				
2.	Run: CALL FindNullHours("ubuntu@ip-172-31-48-19.ec2.internal"); Record COMMIT_ID column for future test expected result.	Non-empty table as a result. No errors producing table.				
3.	Run: SELECT c.commit_ID FROM Commit_info c INNER JOIN Author a USING(author_ID) WHERE a.author_email='ubuntu@ip-172-31-48-19.ec2.internal' AND c.labor_hours IS NULL;	Results in table produced is the same as results recorded in step 1.1.2		· 3.1.1		

1.2 ‘InsertCommitInfo’ Procedure Unit Test

Purpose of Test: Verify accurate functionality of ‘InsertCommitInfo’ procedure.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with MySQL Workbench. **Environment Prerequisites:** MySQL Workbench with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	Run: CALL InsertCommitInfo('2019-11-25 00:20:17',NULL,11, '92f4481ff492f313315932068282120f141282db');	No output				
2.	Run: SELECT * FROM Commit_info;	Entry with last parameter from step 1.2.1 is present				

1.3 ‘InsertCommitTags’ Procedure Unit Test

Purpose of Test: Verify accurate functionality of ‘InsertCommitTags’ procedure.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with MySQL Workbench. **Environment Prerequisites:** MySQL Workbench with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	Run: CALL InsertCommitTags('92f4481ff492f313315932068282120f141282db', 'metrics,v1,v2,v3', 'https://github.com/penatem1/ScriptTesting.git', '3');	Runs without error				
2.	Run: SELECT * FROM Tag INNER JOIN Commit_info USING (commit_ID) WHERE commit_hash='92f4481ff492f313315932068282120f141282db';	Returns table with 3 entries, one for each value in the csv 2nd parameter from step 1.3.1		· 2.3.5		

1.4 ‘InsertAuthor’ Function Unit Test

Purpose of Test: Verify accurate functionality of InsertAuthor’ procedure.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with MySQL Workbench. **Environment Prerequisites:** MySQL Workbench with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	Run: SELECT InsertAuthor('ubuntu@ip-172-31-48-19.ec2.internal', 'Ubuntu');	11				
2.	Run: SELECT InsertAuthor('newEmail@emails.com', 'NewDeveloper');	12				
3.	Run: SELECT InsertAuthor('newEmail@emails.com', 'NewDeveloper');	12				
4.	Run: SELECT * FROM Author WHERE author_ID=12;	Row displays with email ‘newEmail@emails.com’ and name ‘NewDeveloper’		· 2.3.5		

1.5 ‘InsertFile’ Function Unit Test

Purpose of Test: Verify accurate functionality of ‘InsertFile’ function.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with MySQL Workbench. **Environment Prerequisites:** MySQL Workbench with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	Run: SELECT InsertFile('.//src/main.cpp', 'https://github.com/penatem1/ScriptTesting.git');	618				
2.	Run: SELECT InsertFile('.//src/main_new.cpp', 'https://github.com/penatem1/ScriptTesting.git');	652				
3.	Run: SELECT InsertFile('.//src/main_new.cpp', 'https://github.com/penatem1/ScriptTesting.git');	652		· Requirement added during development. (See traceability matrix of validation document.)		

2. GIT HOOK TESTING

2.1 UCC Unit Test

Purpose of Test: Verify functionality of UCC and data parsing for database metrics.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal. **Environment Prerequisites:** Bash environment with UCC, gen_ucc_xml.sh in project directory.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	Run gen_ucc_xml.sh	No output				
2.	View contents of metrics directory	TOTAL_outfile.csv TEMP_TOTAL_outfile.csv NEW_TOTAL_outfile.csv Outfile_maintainability_index.csv TEMP_outfile_maintainability_index.csv NEW_outfile_maintainability_index.csv UCC.xml UCC_main.xml FILES ARE PRESENT		· 2.2.3		
3.	Check contents of NEW_TOTAL_outfile.csv	5 columns in NEW_TOTAL_outfile.csv, with no other data besides csv metrics		· 2.2.3.1		
4.	Check contents of NEW_outfile_maintainability_index.csv	3 columns in NEW_outfile_maintainability_index.csv, with no other data besides csv metrics		· 2.2.3.2		
5.	Check contents of UCC.xml	Valid xml layout				
6.	Check contents of UCC_main.xml	Valid xml layout				

2.2 CppCheck Unit Test

Purpose of Test: Verify functionality of CppCheck for database metrics.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal.
	Environment Prerequisites: Bash environment with cmake, cppcheck, cmake project present in working directory.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	In terminal run: cmake -DCMAKE_EXPORT_COMPILE_COMMANDS=ON	-- Configuring done -- Generating done -- Build files have been written to: $PWD				
2.	In terminal run: cppcheck --xml --project=compile_commands.json --output-file=./cppCheck.xml	Checks all files of project		· 2.2.2		
3.	View contents of cppcheck.xml	Contents of cppcheck.xml are valid		· 2.2.2.1		

2.3 Commit Info Unit Test

Purpose of Test: Verify functionality of commit info aggregation.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal. **Environment Prerequisites:** Git commit is ready to be pushed. Bash environment. Working directory is a git project.
NOTES: Repository is given, Git commit hash and date are given.	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	In terminal run: git log -1 --pretty=format:'%an'	Ubuntu		· 1.1.6		
2.	In terminal run: git log -1 --pretty=format:'%ae'	ubuntu@ip-172-31-48-19.ec2.internal		· 1.1.6		
3.	In terminal run: git log -1 --pretty=format:'%H'	Hash of current commit		· 1.1.5		
4.	In terminal run: git show -s --format=%ci | cut -d' ' --complement -f3	Date of the creation of the commit		· 1.1.5		
5.	In terminal run: cut -d "@" -f 2 <<< $(git config --get remote.origin.url)	Repo url of the testing git project		· 1.1.5		
6.	In terminal run: tr -s '\r\n' ',' <<< $(git tag -l --points-at HEAD) \| sed -e 's/,$/\n/'	alpha,metrics,v1 [NO SPACES]		· 1.1.7		
7.	In terminal run: awk -F"," '{print NF}' <<< "alpha,metrics,v1"	3		· 1.1.7		

2.4 Git Hook Integration Test

Purpose of Test: Verify functionality of the git hook and communication with the database.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal, MySQL, git
	Environment Prerequisites: Git commit ready to be pushed. Commit contains Cppcheck errors. MySQL with database connection.
NOTES: Commit hash, author, and date provided.	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	In terminal run: git tag -f metrics	Completes successfully		· 0.4.1 · 0.4.6		
2.	Git push commit to repository	Pushes successfully				
3.	With mysql run: SELECT * FROM Commit_info INNER JOIN Author USING(author_ID); Record commit_ID of commit hash for futures test validation.	Entry with commit hash is present with commit date and commit author		· 1.1.1 · 1.1.3 · 1.4.2.2 · 1.4.2.3 · 2.3.5		
4.	With mysql run: SELECT * FROM Cpp_metrics;	Entries with commit id from step 2.4.3 is present		· 0.4.3 · 1.4.2.6 · 2.1.2 · 2.3.2		
5.	With mysql run: SELECT * FROM Ucc_metrics;	Entries with commit id from step 2.4.3 is present		· 0.4.5 · 1.4.2.5 · 2.1.3 · 2.1.3.1 · 2.3.1 · 2.4.2		
6.	With mysql run: SELECT * FROM Tag;	Entries with commit id from step 2.4.3 is present. (metrics tag)		· 1.1.8		

3. LABOR HOURS ENTRY TESTING

3.1 Commit ID Input Unit Test

Purpose of Test: Given a user email, verify commits without hours are returned.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal, MySQL **Environment Prerequisites:** Bash environment, MySQL with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	In terminal run: mysql -u root -p'tortoise101$' -h "ec2-18-209-59-108.compute-1.amazonaws.com" -s -N --execute="SELECT c.commit_ID FROM Commit_info c JOIN Author a USING(author_ID) WHERE a.author_email = 'ubuntu@ip-172-31-48-19.ec2.internal' AND c.labor_hours IS NULL" -D Program_info	13 19 22 23 24 25 26 27 28 29 30 [MORE VALUES ALLOWED]		· 3.1.1 · 0.4.5		
2.	In terminal run: mysql -u root -p'tortoise101$' -h "ec2-18-209-59-108.compute-1.amazonaws.com" -s -N --execute="SELECT c.commit_ID FROM Commit_info c JOIN Author a USING(author_ID) WHERE a.author_email = 'bad_email@not_an_email.com' AND c.labor_hours IS NULL" -D Program_info	[NO RETURN]		· 3.1.1		

3.2 Labor Hours Integration Test

Purpose of Test: Verify functionality of the labor hours entry program.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with bash terminal, MySQL Workbench. **Environment Prerequisites:** Bash environment with labor_hours in working directory, MySQL with database connection.
NOTES:	

TEST SCRIPT STEPS / RESULTS

STEP	TEST INPUT	EXPECTED RESULTS	ACTUAL RESULTS	REQUIREMENTS VALIDATED	PASS/FAIL	INIT
1.	In terminal run: labor_hours	Get prompt for email		· 0.4.7		
2.	Enter email that is not associated with a commit	Get reprompt				
3.	Enter email associated with commit	Get prompt for commit_id		· 3.1.1		
4.	Enter invalid commit id (not in list)	Get reprompt				
5.	Enter valid commit id from list	Get hours prompt		· 3.1.2		
6.	Enter invalid hours entry	Get reprompt				
7	Enter valid hours entry	labor_hours finishes		· 3.1.2		
8.	With mysql: query Commit_info for id from step 5	Value is what was input for step 5		· 3.2.1		

4. JENKINS BUILD EVENT TESTING

4.1 Jenkins Build Event Integration Test

Purpose of Test: Verify Jenkins interaction with the MySQL database for updating build status.	
TEST RUN INFO: Tester Name:_____ Test Date:__/__/____	**Tester Prerequisites:** User familiar with Jenkins, MySQL Workbench. **Environment Prerequisites:** Web Browser, Jenkins (prepared with git plugin, build event script, build instructions for a project), MySQL Workbench.
NOTES:	

TEST SCRIPT STEPS / RESULTS						
STEP	**TEST INPUT**	**EXPECTED RESULTS**	**ACTUAL RESULTS**	**REQUIREMENTS VALIDATED**	**PASS/FAIL**	**INIT**
1.	Visit http://ec2-18-209-59-108.compute-1.amazonaws.com:8080/ Login.	Jenkins Dashboard loads.				
2.	Select project. Select 'Build Now'	Build results in build history field.				
3.	In MySQL Workbench run; select * from Program_info. Commit_info;	The 'build_result' field will be updated to a non-null value.		· 1.4.2.7 · 2.3.6		