**Extra Exercise Solutions**

**PL/SQL Extra Exercise 1: CalculateAge Function**

CREATE OR REPLACE FUNCTION CalculateAge(p\_dob DATE)  
RETURN NUMBER IS  
 v\_age NUMBER;  
BEGIN  
 v\_age := TRUNC(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12);  
 RETURN v\_age;  
END;

**Output: Returns the age in years as NUMBER**

**PL/SQL Extra Exercise 2: CalculateMonthlyInstallment Function**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment(p\_loan\_amount NUMBER, p\_annual\_rate NUMBER, p\_duration\_years NUMBER)  
RETURN NUMBER IS  
 v\_monthly\_rate NUMBER;  
 v\_months NUMBER;  
 v\_emi NUMBER;  
BEGIN  
 v\_monthly\_rate := p\_annual\_rate / 12 / 100;  
 v\_months := p\_duration\_years \* 12;  
 v\_emi := (p\_loan\_amount \* v\_monthly\_rate) / (1 - POWER(1 + v\_monthly\_rate, -v\_months));  
 RETURN ROUND(v\_emi, 2);  
END;

**Output: Returns the calculated EMI as NUMBER**

**PL/SQL Extra Exercise 3: HasSufficientBalance Function**

CREATE OR REPLACE FUNCTION HasSufficientBalance(p\_account\_id NUMBER, p\_amount NUMBER)  
RETURN BOOLEAN IS  
 v\_balance NUMBER;  
BEGIN  
 SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_account\_id;  
 RETURN v\_balance >= p\_amount;  
EXCEPTION  
 WHEN NO\_DATA\_FOUND THEN  
 RETURN FALSE;  
END;

**Output: Returns TRUE if balance is sufficient, FALSE otherwise**

**JUnit Extra Exercise: Parameterized Test**

import org.junit.jupiter.params.ParameterizedTest;  
import org.junit.jupiter.params.provider.ValueSource;  
import static org.junit.jupiter.api.Assertions.assertTrue;  
  
public class NumberTest {  
  
 @ParameterizedTest  
 @ValueSource(ints = {2, 4, 6, 8})  
 void testEvenNumbers(int number) {  
 assertTrue(number % 2 == 0);  
 }  
}

**Output: Test passed for all even numbers in the list**

**Mockito Extra Exercise: Argument Matching**

import org.junit.jupiter.api.Test;  
import static org.mockito.Mockito.\*;  
import static org.mockito.ArgumentMatchers.\*;  
  
public class ArgumentMatcherTest {  
  
 @Test  
 void testWithArgumentMatcher() {  
 ExternalApi api = mock(ExternalApi.class);  
 when(api.getData(anyString())).thenReturn("Matched");  
  
 String result = api.getData("anything");  
 assert(result.equals("Matched"));  
 }  
}

**Output: Mocked method returns 'Matched' when any String is passed**