# Sample Weekly Sprint

This document provides a list of things the team can aim to complete at the end of the stated weeks. Depending on the progress of the team, you should adjust your sprints accordingly.

**It is okay to not follow the sample sprints strictly**; team members’ schedule may be affected due to commitments in other modules.

## Legend

[G] General

[SP] Source Processor

[PKB] PKB

[QP] Query Parser

[QE] Query Evaluator

[T] System Testing

## Week 2

[G] Set up tools

[G] Come up with a development plan for Iteration 1

[G] Decide on coding standards

[SP/QP] Research on parsing algorithms

[PKB/QE] Brainstorm on how to store information for their respective components

[T] Set up testing frameworks, scripts etc.

[T] Start writing system test cases (both SIMPLE source code and PQL queries) for the first feature aimed to be developed in the plan

## Week 3

[G] Decide on API documentation standards and correspondence between abstract API types with C++ classes

[SP/PKB/QE] Design PKB APIs

[QP/QE] Design Query Object class

[T] Come up with more test cases according to the features to be implemented in the plan. Test cases generation should be ahead of the schedule by one week (e.g. System test cases for features aimed to be completed in Week 4 must already be generated by end of Week 3)

[G] Fill up Scope, Development Plan, Coding & Documentation Standards, API Design sections in the report

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 3:**

1. Parsing and validation of read/print statement
2. Population of design entity information such as variable, statement number, procedure
3. Parsing of queries with no such-that/pattern clauses
4. Evaluation of queries with no such-that/pattern clauses, focusing on selection of all possible synonyms correctness

## Week 4

[G] Fill up Testing section in the report

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 4:**

1. Parsing and validation of assign statement
2. Population of F/F\* relationship information
3. Parsing of queries with one such-that clause or one pattern clause
4. Evaluation of queries with one clause, focusing on F/F\* correctness

## Week 5

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 5:**

1. Parsing and validation of while/if statements
2. Population of P/P\* relationship and pattern information
3. Parsing of queries with one such-that clause and one pattern clause
4. Evaluation of queries with one clause, focusing on P/P\*/pattern correctness

## Week 6 [Iteration 1 code submission: Recess Week Mon]

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 6:**

1. Population of M/U relationship information
2. Parsing of queries with one such-that clause and one pattern clause
3. Evaluation of queries with two clauses, focusing on M/U and table merging correctness

## Recess Week

[G] Complete Iteration 1 report draft

## Week 7

[G] Come up with a development plan for Iteration 2 and 3

[G] Fix bugs found in Iteration 1

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 7:**

1. Parsing and validation of call statements & multiple procedures
2. Population of full assign and container pattern information
3. Parsing of queries with selection of BOOLEAN, other additional synonyms such as prog\_line, call etc., new pattern clauses
4. Evaluation of queries, focusing on BOOLEAN, F/F\*/P/P\*/M/U and pattern clause correctness

## Week 8

[G] Fill up Scope, Development Plan, Testing, Coding & Documentation Standards, API Design sections in the report

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 8:**

1. Parsing and validation of call statements & multiple procedures
2. Population of C/C\*, and full M/U relationship information
3. Parsing of queries with multiple clauses and and-keyword
4. Evaluation of queries with multiple clauses, focusing on C/C\*/M/U and table merging correctness

## Week 9 [Iteration 2 report submission: Week 10 Mon]

[G] Complete Iteration 2 report

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 9:**

1. Population of N relationship and with-clause information
2. Parsing of queries with with-clauses and attributes
3. Evaluation of queries with multiple clauses, focusing on N relationship, with-clauses, synonym attributes and table merging correctness

## Week 10 [Iteration 2 report submission: Week 10 Mon, Iteration 2 demo: Week 10 Tue/Wed]

[T] Make sure system is well-tested by adding new test cases that are found missing in the demo

[SP/QP] Help out with test cases, or N\*/A/A\* generation, or optimisation

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 10:**

1. Population of N\*/A relationship in query-time
2. Parsing of queries with tuples
3. Evaluation of queries with multiple clauses, focusing on N\*/A relationship, tuple, synonym attributes and table merging correctness

## Week 11

[G] Fill up Scope, Development Plan, Testing, Coding & Documentation Standards, API Design sections in the report

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 11:**

1. Population of A\* relationship in query-time
2. Parsing of clauses to facilitate optimisation
3. Evaluation of queries with optimisation, focusing on A\* correctness and finding out which optimisation strategy works best

## Week 12

[G] Complete Iteration 3 Report Draft

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 12:**

1. Population of NB/NB\* relationship extension
2. Rearrange clauses for optimisation
3. Evaluation of queries with optimisation, focusing on base system correctness and optimisation strategies, refinement

## Week 13 [Iteration 3 code + report submission: Week 13 Fri]

[G] Conduct presentation rehearsal after code + report submission

**Features to be implemented and well-tested with Unit/Integration/System Tests by Week 13:**

1. Population of AB/AB\* relationship extension
2. Evaluation of queries with optimisation, focusing on extension correctness