**CQP log Checking – Design Note #1**

**Version 1.0**

**September 1st. 2012**

This note is a working document to capture the SQL tables needed for processing CQP logs from submission through scoring.

**SQL Required Tables**

The following sections describe the tables to be used at each step in the log processing. Process steps are described in the document “CQP 2012 Log Processing.docx” available in the GIT repository.

Process steps requiring database support are:

2.0 (h) Capture of header information

3.0 Normalization

4.0 Log Record Matching

5.0 Scoring

6.0 Result Generation

**Capture of header information**

A table is required to reference all participants in the contest and serves as the master data for tracking entries, entry categories, and scoring information. Population of data in this table will originate from two sources – automated processing of submitted logs and manual updates to provide missing information or information contained in the soapbox comments.

CREATE TABLE cqp(

pk\_cqp INTEGER PRIMARY KEY AUTOINCREMENT,

call VARCHAR(16), // Call sign of entrant

qth VARCHAR(4), // QTH of entrant

category VARCHAR(2), // Operating category

power CHAR, // Power level

in\_ca BOOL, // In California?

youth BOOL, // Youth category

yl BOOL, // YL category

mobile BOOL, // Mobile category

cce BOOL, // County expedition

newc BOOL, // New Contester category

school BOOL, // School category

p\_claimed INTEGER, // Claimed score

cw\_made INTEGER, // Number of CW QSOs made

ph\_made INTEGER, // Number of phone QSOs made

m\_made INTEGER, // Number of multipliers made

score INTEGER, // Log checked score,

t\_58 TIME, // Time to 58

);

The clubs table has a single entry for each unique club participating in the club contest together with an Boolean indicating whether the club is in California or not. This is the master list of all unique club names for which entrants reference in their entries.

The table is populated with known clubs prior to the contest and then manually updated for new clubs as they appear.

CREATE TABLE clubs(

pk\_clubs VARCHAR(10) PRIMARY KEY, // Club ID for CQP

name VARCHAR(60), // Name of club

in\_ca BOOL, // California club

);

The club\_alloc table is used to apportion scores to clubs. In most cases there will be a one to one entry that allocates a total score to a single club. However, rules permit multi-operator stations to apportion their score based on the membership of the operators.

For example, Ni6T as a M/M may allocate 20% of its score to REDXA. This information gets captured via the web submission form or is noted in the SOAPBOX comments. An entry would be present in the club\_alloc table for each club allocation of a score together with the percentage to be applied.

As a sanity check, the sum of all entries for a specific combination of fk\_clubs and fk\_cqp should be 100 or less.

This table is auto-populated by script processing where possible and then updated manually by contest administration.

CREATE TABLE club\_alloc(

pk\_club\_alloc INTEGER PRIMARY KEY AUTOINCREMENT,

FOREIGN KEY (fk\_clubs) REFERENCES clubs(pk\_clubs),

FOREIGN KEY (fk\_cqp) REFERENCES cqp(pk\_cqp),

alloc INTEGER, // Percentage allocated

);

**Normalization**

Normalization of logs is split into two parts; header record normalization and QSO record normalization.

**Header Records (what a can of worms…)**

Incoming files are a mixture of Cabrillo versions 2 and 3 and must be normalized to a common format (version 3 which is a superset of 2). As a note, 2/3rd of the logs submitted in 2011 where in 2.0 format.

Version 2.0 processing: Information regarding category and power are contained on one or more CATEGORY lines. Regardless of the spec, there may be one or multiple CATEGORY lines containing the information and may or not be in Cabrillo standard form.

The logic to map headers to a single version is mostly contained in two scripts:

Cabrillo.pm: A perl class written by Try N5KO to manipulate a Cabrillo log according to the whether it is in version 2 or 3 in a version 3 compliant interface.

prep\_extractor.pl: A perl script using the above class library to change headers into a CQP compliant format.

A review of the logs from 2011 suggest that neither of these scripts are exhaustive as many loggers place information in these fields that are not Cabrillo compliant – for example; SO-LP, S0-HP etc.

We need a uniform script to post process all headers or to flag a log with none compliant information.