WisCALM 2022 – Lake Trophic State Index (TSI) Assessment Parameter Documentation

EGAD # 3200-2020-32

A product of the State of Wisconsin Clean
Water Act Water Quality Report to Congress

Red Cedar River, Wisconsin DNR

Ashley Beranek, Project Manager
Macaulay Haller, Project Technician
Brian Tinberg, Systems Architect
Jacob Dickmann, SWIMS Database File Manager
Will Westbury, WATERS Database File Manager

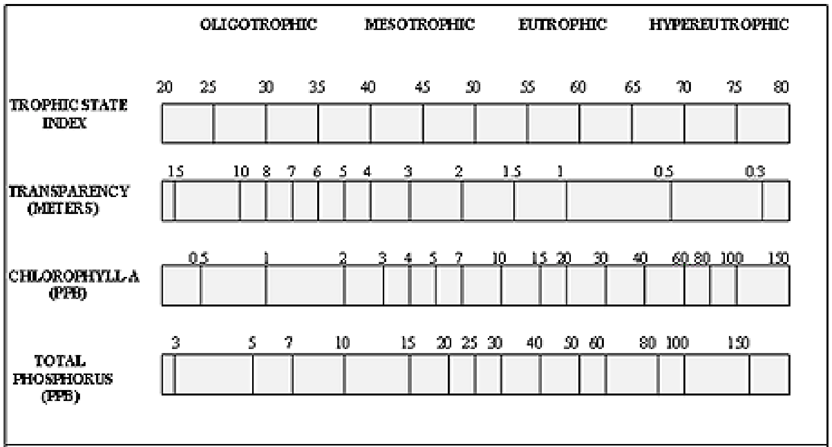
Contents

Parameter Names and Numbers

Lake 10 Year TSI Chla Assessment Value	80423
Lake 10 Year TSI TP Assessment Value	80424
Lake 10 Year TSI Secchi Assessment value	80425
Lake 10 Year TSI Satellite Secchi Assessment Value	80426

Description

Wisconsin bases its General Condition Assessment for lakes on the Carlson Trophic State Index (TSI). The Carlson TSI is the most commonly used index of lake productivity. It provides separate, but relatively equivalent, TSI calculations based on either chlorophyll *a* concentration (chlorophyll *a*, or CHL in the equation below) or Secchi depth (SD, for which Wisconsin also uses satellite clarity data as a surrogate)[1]. Because TSI is a prediction of algal biomass, typically the chlorophyll *a* value is a better predictor than Secchi or satellite data. Water clarity as measured by Secchi depth or satellite is a practical measure of algal production and water color. Algal production is known to be highly correlated with nutrient levels (especially phosphorus). High levels of nutrients can lead to eutrophication and blue-green algae blooms. This limits the amount of available light to macrophytes and adversely affects other aquatic organisms. Information from each of these parameters is valuable because the interrelationships between them can be used to identify other environmental factors that may influence algal biomass.



After Moore, L. and K. Thornton, [Ed.]1988. Lake and Reservoir Restoration Guidance Manual. USEPA>EPA 440/5-88-002.

Figure 1. Continuum of lake trophic status in relation to Carlson Trophic State Index.

[1] Carlson also provides an equation to convert total phosphorus concentration to TSI, but WDNR is not using that equation for purposes of water quality assessments or 303(d) Impaired Waters Listing.

TSI values range from low (less than 30), representing very clear, nutrient-poor lakes, to high (greater than 70) for extremely productive, nutrient-rich lakes. Very few lakes in Wisconsin would fall into the category of "very clear, nutrient poor lakes." The cutoff for excellent TSI values would certainly include these lakes, but also includes some lakes in the mesotrophic category, based on sediment core data which indicates that some lakes are naturally more productive than others.

This effort has been built on a successful collaboration between UW-Madison, WDNR and the Citizen Lake Monitoring Network. Landsat satellite imagery is used in conjunction with citizen-collected Secchi depths to develop models that estimate water clarity in lakes > 5 acres statewide. This WDNR-Science Services activity, performed annually, now has 25 years of record. At least two water clarity values from within a 3-year period in summer are averaged to determine lake trophic status.

Data Sources and Storage

The following parameters are collected by DNR staff, volunteers, and by members of other organizations: total phosphorus, chlorophyll a, and secchi depth. These data are collected in the field as per the protocols listed below and then analyzed at certified laboratories, typically the State Lab of Hygiene (SLOH). The results are then loaded from the labs into the SWIMS database. Secchi-satellite images and measurements are collected from Landsat Satellites 7 and 8 and are archived on DNR systems.

Methods and procedures to document and store

- Lake Sampling Procedures – LTT Water Quality
- Landsat 7 and 8 Satellite Monitoring Schedule 2020
- Wisconsin Citizen Lake Monitoring Secchi Disk Procedures
- Wisconsin Citizen Lake Monitoring Chemistry Procedures

Data Entry

Total phosphorus and chlorophyll *a* data analyzed by the SLOH are sent to SWIMS via the Lab Data Entry System. If connections are established with other laboratories, data can be sent to SWIMS via those connections. Secchi measurements are typically entered through the SWIMS interface. A spreadsheet batch upload process may be utilized to enter the above parameters to SWIMS. All records and image files for Secchi-satellite are archived on DNR systems. A file containing the secchi estimates is sent annually to the lakes program and stored in SWIMS.

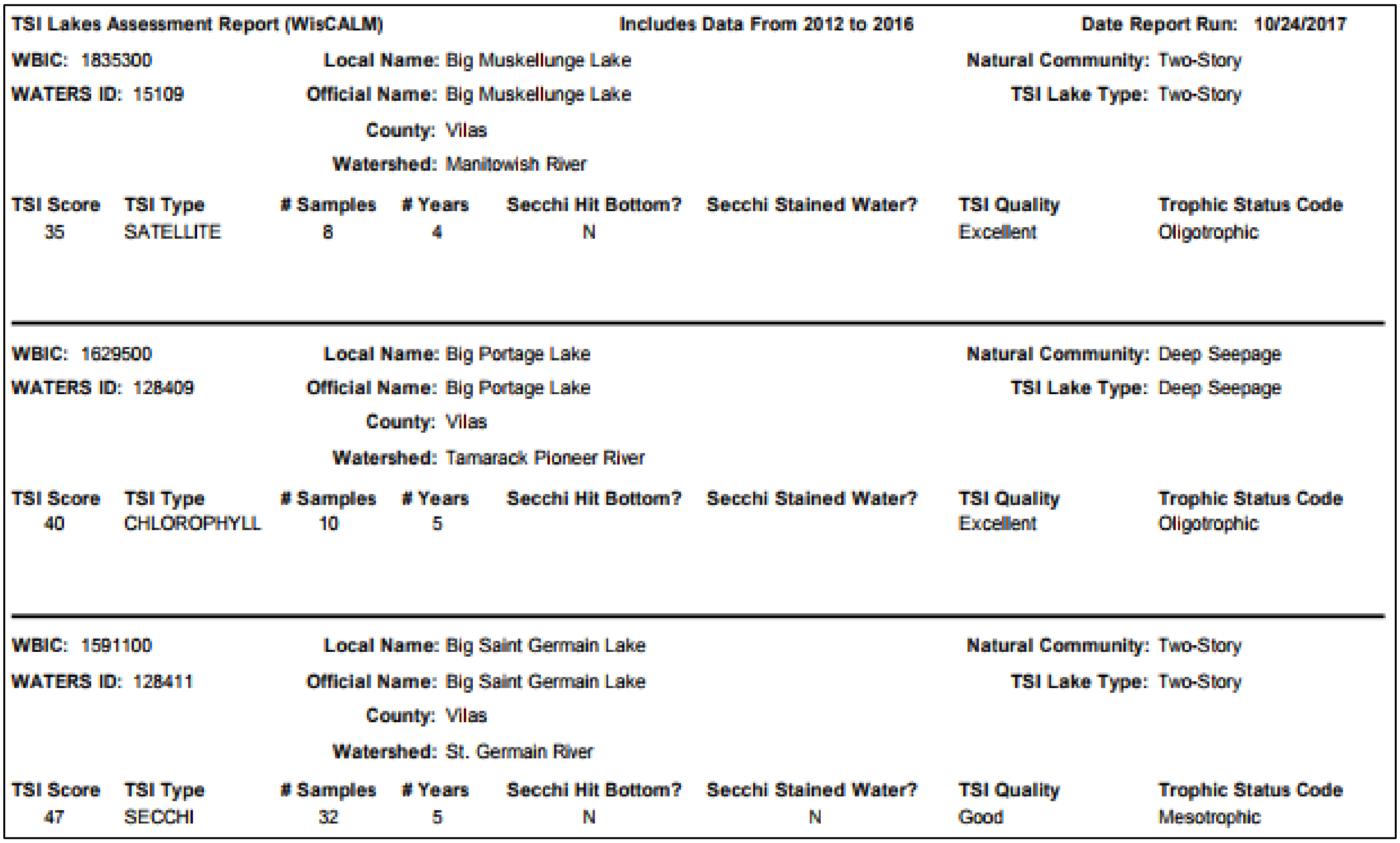
Presentation of Results

Presentation of Results in WATERS online report and Water Condition Viewer

How do the new parameters fit into the existing multi-part assessment categorization process?

- The parameter may be assessed independently and in conjunction with lake chlorophyll and aquatic plant biocriteria, and may affect the waterbody assessment categorization for the FAL and REC use assessments.

View of WATERS online report for TSI



TSI Lakes Assessment Report (WisCALM) **Includes Data From 2012 to 2016** **Date Report Run:** 10/24/2017

WBIC: 1835300 **Local Name:** Big Muskellunge Lake **Natural Community:** Two-Story
WATERS ID: 15109 **Official Name:** Big Muskellunge Lake **TSI Lake Type:** Two-Story
County: Vilas
Watershed: Manitowish River

TSI Score	TSI Type	# Samples	# Years	Secchi Hit Bottom?	Secchi Stained Water?	TSI Quality	Trophic Status Code
35	SATELLITE	8	4	N		Excellent	Oligotrophic

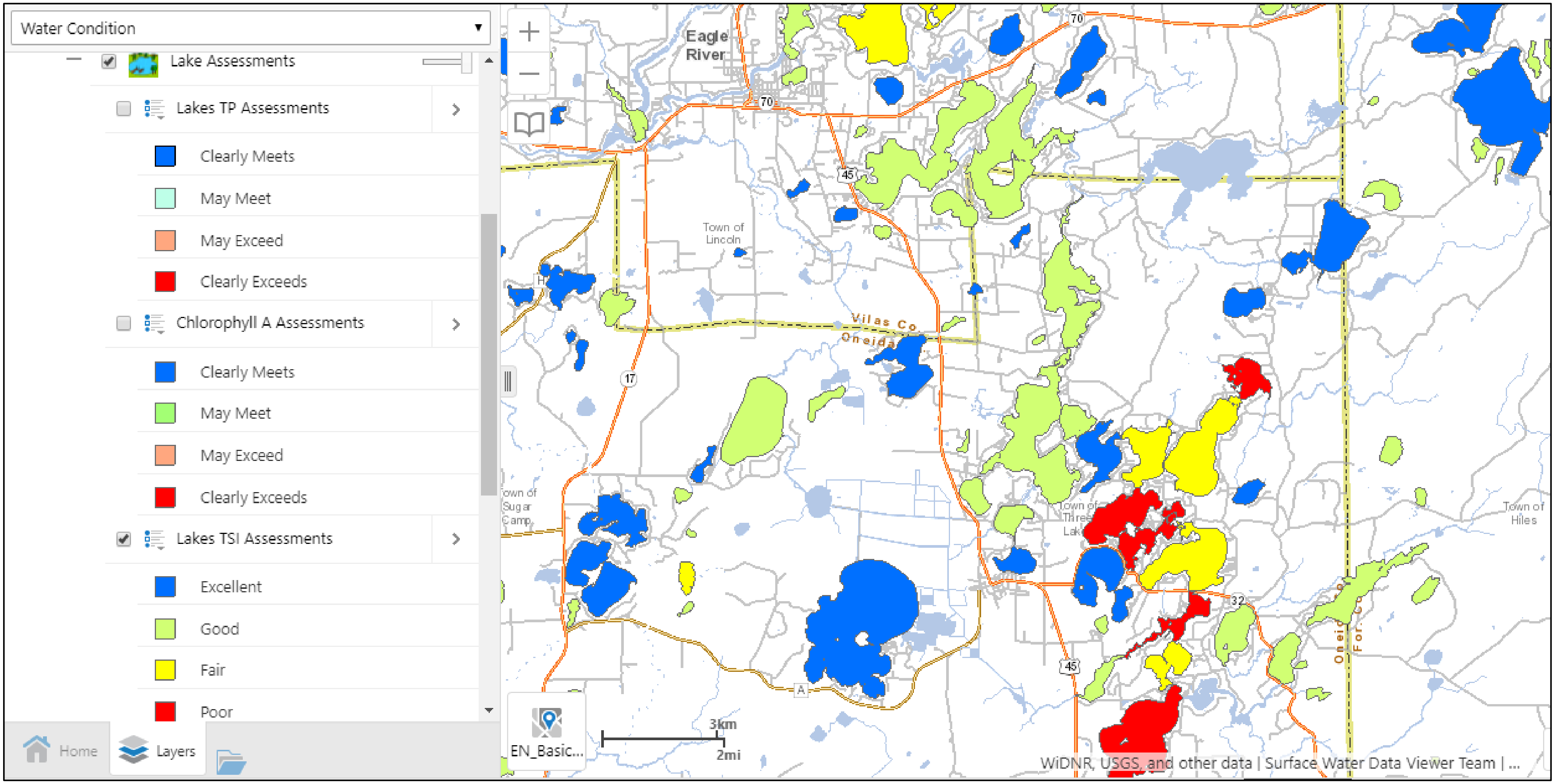
WBIC: 1629500 **Local Name:** Big Portage Lake **Natural Community:** Deep Seepage
WATERS ID: 128409 **Official Name:** Big Portage Lake **TSI Lake Type:** Deep Seepage
County: Vilas
Watershed: Tamarack Pioneer River

TSI Score	TSI Type	# Samples	# Years	Secchi Hit Bottom?	Secchi Stained Water?	TSI Quality	Trophic Status Code
40	CHLOROPHYLL	10	5			Excellent	Oligotrophic

WBIC: 1591100 **Local Name:** Big Saint Germain Lake **Natural Community:** Two-Story
WATERS ID: 128411 **Official Name:** Big Saint Germain Lake **TSI Lake Type:** Two-Story
County: Vilas
Watershed: St. Germain River

TSI Score	TSI Type	# Samples	# Years	Secchi Hit Bottom?	Secchi Stained Water?	TSI Quality	Trophic Status Code
47	SECCHI	32	5	N	N	Good	Mesotrophic

View of Water Condition Viewer report for TSI



Water Condition ▾

☑ Lake Assessments

☐ Lakes TP Assessments ›

🟦 Clearly Meets

🟩 May Meet

🟧 May Exceed

🟥 Clearly Exceeds

☐ Chlorophyll A Assessments ›

🟦 Clearly Meets

🟩 May Meet

🟧 May Exceed

🟥 Clearly Exceeds

☑ Lakes TSI Assessments ›

🟦 Excellent

🟩 Good

🟨 Fair

🟥 Poor

Assessment Package Logic

Lakes TSI Assessment Package Rules:

Based on 2018 WisCALM assessment method for assessment of Lake TSI using secchi (field and satellite), chlorophyll *a*, and total phosphorus data

Prepared by: Ashley Beranek
August 18, 2016

Updated by: Ashley Beranek, Jake Dickmann, Lisa Helmuth, Brian Tinberg
January 29, 2018

Oracle package populates table, which includes:

- TSI score
- Which TSI type was used: TSI(chlorophyll), TSI(Secchi) or TSI(Secchi-Satellite)
- WBIC
- Number of samples included in average
- Number of years included
- Secchi hit bottom?
- Stained water?
- Below level of detection?
- Standard deviation (?)

Additional tables may summarize TSI by station, year and/or WBIC for informational purposes.

All

- Year should be within last 5 years only. Include the current year starting 1/1 of the following year. So, starting on 1/1/2009, include 2008 results.
- Ignore sample if QC_FLAG <> 1 (this will eliminate blanks, duplicates and data we flag as being "bad data" such as extreme Secchi outliers).

TSI (Secchi)

- Carlson TSI equation
- Fieldwork Start Date must be between 7/15-9/15
- A minimum of 2 samples / year
- If there is at least 1 sample header included that also has DNR_STORET parameter 99420 result = "Y" or "YES", populate "Secchi Hit Bottom" column with "Yes"
- If there is at least 1 sample header included that also has SWIMS parameter 90000 result = CLEAR and SWIMS parameter 90001 result = BROWN OR RED OR YELLOW: populate "Stained Water?" column with "Yes"
- Secondary Station Type should be DEEPEST SPOT

TSI (Secchi-Satellite)

- Secondary Station Type should be null & Feature Type should be AREA.
- Fieldwork Start Date must be between 7/15-9/30 (allow longer window for satellite data)
- A minimum of 1 sample / year

- If there is at least 1 sample header included that also has SWIMS parameter 90943 and 90942 result = “Y” or “YES”, populate “SAT_SECCHI_HIT_BOTTOM_FLAG” column with “Yes”
- Secchi-satellite data obtained and uploaded to SWIMS through backend processes. These steps are outlined as follows:

  - Integrated Reporting staff (Program staff) acquire secchi-satellite data from Lakes and River Section (LRS) staff
    - Secchi-satellite data are obtained, prepared, and analyzed by research staff. The results are sent to LRS staff. This is done annually
  - Secchi-satellite data are loaded into a SWIMS worktable on SECPRD
    - A SWIMS View adds the following items to the raw data:
      - i. The monit_station_seq_no (if it can be determined):
        - A PL/SQL function is used in attempt to fill the station for the correct waterbody via this process:
          - i. Uses the WBIC to find exactly one matching station from wt_swims_monit_sta_isect_gv. If zero or greater than one are found then it moves on to ii
          - ii. Uses the HYDROID to find exactly one matching station from wt_swims_monit_sta_isect_gv. If zero or greater than one are found then it moves on to iii
          - iii. Lookup the station in the table brian_sat_secchi_station via WBIC (this is typically done manually by Program or LRS staff)
        - In some cases, the function is unable to determine a station. The following query provides a list of those waterbodies:

          ```
          select *
            from w07510.brian_2016_sat_secchi_v
          where monit_station_seq_no is null and
          coalesce(row_waterbody_type_code, '~') <> 'ST';
          ```

          - i. Once those waterbodies are determined, Program or LRS staff search for the correct station to list, create a new station if needed (type of station = AREA), or determine that the waterbody is not acceptable to assess (the waterbody is not located in Wisconsin, i.e. Mississippi River backwaters located in Minnesota, etc.)
      - ii. A hit_bottom_flag, which gets set to “Y” if the lake depth from the Register of Waterbodies (ROW) database is less than the satellite secchi depth (sd_mean).
      - iii. Waterbody type and max depth from ROW
  - Once stations are assigned, a project in SWIMS is created to house the satellite secchi data
    - Fieldwork events, sample headers, and sample results are created for each and loaded into the project

TSI (Chlorophyll)

- Carlson TSI equation
- Fieldwork Start Date must be between 7/15-9/15
- A minimum of 2 samples / year
- If below level of detection (result_value_no contains ND), populate “Below Level of Detection” column with “Yes”. And use ½ of LOD as the result_amt.

- Vertical measure start & end depth must be < 7 feet or < 2 meters. Cannot be null.
- Secondary Station Type should be DEEPEST SPOT

TSI (Total Phosphorus)

- Carlson TSI equation
- Fieldwork Start Date must be between 7/15-9/15
- A minimum of 2 samples / year
- If below level of detection (result_value_no contains ND), populate "Below Level of Detection" column with "Yes". And use ½ of LOD as the result_amt.
- Vertical measure start & end depth must be < 7 feet or < 2 meters. Cannot be null. If more than one, average the results together.
- Secondary Station Type should be DEEPEST SPOT

TSI value for lake as a whole

- Primary station type should be LAKE or RESERVOIR or RIVERINE IMPOUNDMENT
- Pick 1 TSI to use (see hierarchy below)
    - If we have TSI(chlorophyll) for 3 different years, use TSI(Chlorophyll)
    - Else if we have TSI(Secchi) for 3 different years, use TSI(Secchi)
    - Else if we have TSI(satellite) for 3 years, use TSI(satellite)
    - Otherwise no result
- Average together value for each applicable year to get the single TSI.
- Combines SECCHI_HIT_BOTTOM_FLAG and SAT_HIT_BOTTOM_FLAG into one column= SECCHI_HIT_BOTTOM_FLAG

Once all the data are loaded and the proper constraints are applied, the TIS Assessment Package is run (see Assessment Package Code on pg. 9)

How the package performs the calculations:

- Calculates chlorophyll, Secchi and satellite TSI for each station and year combination.
- Averages the above to get chlorophyll, Secchi and satellite TSI for each WBIC and year combination.
- Averages the above to get a single TSI for each WBIC (5 year average, picks the best TSI)

Program staff work with the Database Programmer to update assessment decisions in WATERS based on TSI data:

- **Is water impaired? N – Category 3 waters: batch upload into WATERS** [if it is a category 3 there should not really be an assessment but there could be Cat. 3 by mistake or not enough data]
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it go up in quality: Y → update record(s):
        - date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it stay the same: Y → update record(s)
        - date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment Y → does it go down in quality: Y → review the data and if quality and accurate, update record(s) and make recommendation for more monitoring
        - assessment decision, category, date, methodology, assessment date, data for decision, information sources, xref to document

- **Is water impaired? N – Category 2 waters: update in WATERS by hand**
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it go up in quality: Y → update record(s):
        - date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it stay the same: Y → update record(s):
        - date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it go down in quality: Y → review the data and if quality, update record(s) and make recommendation for more monitoring, perhaps change to "Category 3" again:
        - assessment decision, category, date, methodology, assessment date, data for decision, information sources, xref to document

- **Is water impaired? Y – Category 4 or 5 waters: update in WATERS by hand**
    - Does the lake already have an assessment Y → does this change the assessment: Y → does it go up in quality: Y → update record(s):
        - assessment decision, category, date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it stay the same: Y → update record(s):
        - assessment decision, category, date, methodology, assessment date, data for decision, information sources, xref to document
    - Does the lake already have an assessment: Y → does this change the assessment: Y → does it go down in quality: Y → review the data and if quality, update record(s) and make recommendation for more monitoring, perhaps change to "Category 3" again:
        - assessment decision, category, date, methodology, assessment date, data for decision, information sources, xref to document

For all waters that are not considered impaired, a narrative is placed in the Water Condition Narrative section of WATERS outlining what was assessed and that the results indicate this waterbody is meeting it's assigned water quality criteria. This is dated.

For all waters that are considered impaired, a narrative is placed in the Water Condition Narrative section of WATERS outlining what was assessed and that the results indicate this water body is impaired and is not meeting it's assigned water quality criteria. This is dated.

Assessment Package Code

```
         W07510.pk_swims_tsi is

   procedure p_start_station(i_station in number, i_year in number,
   io_tsi_rec in out wt_swims_tsi_station_year%rowtype) is
   begin
   io_tsi_rec.monit_station_seq_no := i_station;
   io_tsi_rec.tsi_year := i_year;
   io_tsi_rec.tsi_chlor_samp_count := 0;
   io_tsi_rec.tsi_phos_samp_count := 0;
   io_tsi_rec.tsi_secchi_samp_count := 0;
```

```
io_tsi_rec.tsi_sat_secchi_samp_count := 0;
io_tsi_rec.tsi_chlor_score_amt := null;
io_tsi_rec.tsi_phos_score_amt := null;
io_tsi_rec.tsi_secchi_score_amt := null;
io_tsi_rec.tsi_sat_secchi_score_amt := null;
io_tsi_rec.secchi_hit_bottom_flag := 'N';
io_tsi_rec.secchi_stained_water_flag := 'N';
io_tsi_rec.sat_secchi_hit_bottom_flag := 'N';
io_tsi_rec.chlor_below_lod_flag := 'N';
io_tsi_rec.phos_below_lod_flag := 'N';
exception
when others then
      pk_swims_job_log.p_logit('Exception raised in p_start_station: ' ||
      sqlerrm);
raise;
end;

procedure p_end_station(i_tsi_rec in wt_swims_tsi_station_year%rowtype) is
begin
insert into wt_swims_tsi_station_year
      (monit_station_seq_no,
      tsi_year,
      tsi_chlor_samp_count,
      tsi_phos_samp_count,
      tsi_secchi_samp_count,
      tsi_sat_secchi_samp_count,
      tsi_chlor_score_amt,
      tsi_phos_score_amt,
      tsi_secchi_score_amt,
      tsi_sat_secchi_score_amt,
      secchi_hit_bottom_flag,
      secchi_stained_water_flag,
      sat_secchi_hit_bottom_flag,
      chlor_below_lod_flag,
      phos_below_lod_flag)
      values
      (i_tsi_rec.monit_station_seq_no,
      i_tsi_rec.tsi_year,
      i_tsi_rec.tsi_chlor_samp_count,
      i_tsi_rec.tsi_phos_samp_count,
      i_tsi_rec.tsi_secchi_samp_count,
      i_tsi_rec.tsi_sat_secchi_samp_count,
      i_tsi_rec.tsi_chlor_score_amt,
      i_tsi_rec.tsi_phos_score_amt,
      i_tsi_rec.tsi_secchi_score_amt,
      i_tsi_rec.tsi_sat_secchi_score_amt,
      i_tsi_rec.secchi_hit_bottom_flag,
      i_tsi_rec.secchi_stained_water_flag,
      i_tsi_rec.sat_secchi_hit_bottom_flag,
      i_tsi_rec.chlor_below_lod_flag,
      i_tsi_rec.phos_below_lod_flag);
commit;
exception
when others then
      pk_swims_job_log.p_logit('Exception raised in p_end_station: ' ||
      sqlerrm);
raise;
```

```
end;

function f_compute_tsi(i_tsi_type in varchar2, i_result_amt in number)
return number is
begin
            Carlson equations
case i_tsi_type
      when 'SECCHI' then return 60 - (14.41 * ln(i_result_amt));
      when 'SATELLITE' then return 60 - (14.41 * ln(i_result_amt));
      when 'CHLOROPHYLL' then return (9.81 * ln(i_result_amt)) + 30.6;
      when 'PHOSPHORUS' then return (14.42 * ln(i_result_amt)) + 4.15;
end case;
exception
when others then
      pk_swims_job_log.p_logit('Exception raised in f_compute_tsi: ' ||
      sqlerrm);
raise;
end;

function f_check_chlor_phos_depth(i_res_seq in number, i_hdr_seq in
number, i_fw_seq in number) return boolean is
v_depth_ok  boolean := false;
v_vm_found  boolean := true;
v_vert_meas wt_swims_vertical_measure%rowtype;
cursor c_res_vm is select * from wt_swims_vertical_measure where
sample_result_seq_no = i_res_seq order by start_amt;
cursor c_hdr_vm is select * from wt_swims_vertical_measure where
sample_header_seq_no = i_hdr_seq order by start_amt;
cursor c_fw_vm is select * from wt_swims_vertical_measure where
fieldwork_seq_no = i_fw_seq order by start_amt;
begin
            depth must be <= 2 meters or 7 feet
open c_res_vm;
fetch c_res_vm into v_vert_meas;
if c_res_vm%NOTFOUND then
      open c_hdr_vm;
      fetch c_hdr_vm into v_vert_meas;
      if c_hdr_vm%NOTFOUND then
      open c_fw_vm;
      fetch c_fw_vm into v_vert_meas;
      if c_fw_vm%NOTFOUND then
            v_vm_found := false;
      end if;
      close c_fw_vm;
      end if;
      close c_hdr_vm;
end if;
close c_res_vm;
if v_vm_found then
      case v_vert_meas.unit_code
      when 'METERS' then
            if v_vert_meas.start_amt <= 2.0 then
            v_depth_ok := true;
            end if;
      when 'FEET' then
            if v_vert_meas.start_amt <= 7.0 then
            v_depth_ok := true;
            end if;
      when 'CM' then
            if v_vert_meas.start_amt <= 200.0 then
            v_depth_ok := true;
            end if;
      when 'CENTIMETER' then
            if v_vert_meas.start_amt <= 200.0 then
            v_depth_ok := true;
            end if;
      when 'IN' then
            if v_vert_meas.start_amt <= 84.0 then
            v_depth_ok := true;
            end if;
      end case;
end if;
return v_depth_ok;
exception
when others then
      pk_swims_job_log.p_logit('Exception raised in
      f_check_chlor_phos_depth: ' || sqlerrm);
raise;
end;

procedure p_end_tsi_type(i_tsi_type in varchar2, i_tsi_sum in number,
i_tsi_cnt in integer,
                              io_tsi_rec in out
                              wt_swims_tsi_station_year%rowtype) is
v_tsi_avg number;
begin
            this rule will be applied at the WBIC level, not the station
            level.
--if i_tsi_cnt >= 2 then  -- must have at least two readings to get a TSI
score for the type of TSI
--
if i_tsi_cnt >= 1 then
      v_tsi_avg := i_tsi_sum / i_tsi_cnt;
      case i_tsi_type
      when 'CHLOROPHYLL' then
            io_tsi_rec.tsi_chlor_samp_count := i_tsi_cnt;
            io_tsi_rec.tsi_chlor_score_amt := v_tsi_avg;
      when 'PHOSPHORUS' then
            io_tsi_rec.tsi_phos_samp_count := i_tsi_cnt;
            io_tsi_rec.tsi_phos_score_amt := v_tsi_avg;
      when 'SECCHI' then
            io_tsi_rec.tsi_secchi_samp_count := i_tsi_cnt;
            io_tsi_rec.tsi_secchi_score_amt := v_tsi_avg;
      when 'SATELLITE' then
            io_tsi_rec.tsi_sat_secchi_samp_count := i_tsi_cnt;
            io_tsi_rec.tsi_sat_secchi_score_amt := v_tsi_avg;
      end case;
end if;
exception
when others then
      pk_swims_job_log.p_logit('Exception raised in p_end_tsi_type: ' ||
      sqlerrm);
raise;
end;
```

```
procedure p_load_station_year(i_year in number, i_station in number :=
null) is
v_start                date := to_date('0715' || to_char(i_year),
'mmddyyyy');
v_end                  date := to_date('0915' || to_char(i_year),
'mmddyyyy') + (86399 / 86400);
v_sat_end              date := to_date('0930' || to_char(i_year),
'mmddyyyy') + (86399 / 86400); -- look up to 9/30 for satellite data
v_hold_station         number(10) := -1;
v_tsi_rec              wt_swims_tsi_station_year%rowtype;
v_tsi_sum              number;
v_tsi_cnt              number;
v_hold_tsi_type        varchar2(20);
v_hold_date            date;
v_date_sum             number;
v_date_cnt             integer;
v_result_amt           number;
v_depth_ok             boolean;
v_dummy                boolean;
v_insert_cnt           integer := 0;
v_temp_cnt             integer;
begin
v_dummy := pk_swims_job_log.initialize('TSI LOAD');
pk_swims_job_log.p_logit('i_year = ' || i_year || ', i_station = ' ||
nvl(to_char(i_station),'null'));

            remove any existing records for the year (and station if
            provided)
delete from wt_swims_tsi_station_year
      where tsi_year = i_year and
            monit_station_seq_no = nvl(i_station, monit_station_seq_no);
pk_swims_job_log.p_logit('Removed ' || sql%rowcount || ' records from
wt_swims_tsi_station_year.');
commit;

            grab all the relevant results for processing
for cv_res in (select case dnr_parameter_code
                              when 99717 then 'CHLOROPHYLL'
                              when 32211 then 'CHLOROPHYLL'
                              when 665   then 'PHOSPHORUS'
                              when 49701 then 'SECCHI'
                              when 78    then 'SECCHI'
                              when 90880 then 'SATELLITE'
                        end tsi_type,
                        monit_station_seq_no, trunc(result_date_time)
                        result_date, fieldwork_seq_no,
                        sample_result_seq_no, sample_header_seq_no,
                        result_amt, result_units_text, result_value_no,
                        lod_amt, result_qualifier_code
                  from wt_swims_result_fact_v r
                  where ((dnr_parameter_type = 'DNR_STORET' and
                  dnr_parameter_code in (99717,32211,665,49701,78)) or
                        (dnr_parameter_type = 'SWIMS' and
                        dnr_parameter_code = 90880)) and
```

```
                              ((dnr_parameter_code in (99717,32211,665,49701,78)
                              and r.result_date_time between v_start and v_end)
                              or
                              (dnr_parameter_code = 90880 and r.result_date_time
                              between v_start and v_sat_end)) and
                              qc_flag = '1' and
                              monit_station_seq_no = nvl(i_station,
                              monit_station_seq_no) and
                              ((result_amt is not null and result_amt > 0) or
                              result_value_no like '%ND%' or
                              result_qualifier_code = '2')
                        order by monit_station_seq_no, tsi_type, result_date)
                        loop
          if cv_res.monit_station_seq_no <> v_hold_station then
          if v_hold_station <> -1 then
                chlorophyll and phosphorus need to accumulate and average
                values taken on the same date,
                secchi does not
                if v_hold_tsi_type in ('CHLOROPHYLL','PHOSPHORUS') and
                v_date_cnt > 0 then
                v_tsi_sum := v_tsi_sum + f_compute_tsi(v_hold_tsi_type,
                v_date_sum / v_date_cnt);
                v_tsi_cnt := v_tsi_cnt + 1;
                end if;
                p_end_tsi_type(v_hold_tsi_type, v_tsi_sum, v_tsi_cnt,
                v_tsi_rec);
                p_end_station(v_tsi_rec);
                v_insert_cnt := v_insert_cnt + 1;
          end if;
          v_hold_station := cv_res.monit_station_seq_no;
          p_start_station(cv_res.monit_station_seq_no, i_year, v_tsi_rec);
          v_hold_tsi_type := 'XXX';
          end if;

          if cv_res.tsi_type <> v_hold_tsi_type then
          if v_hold_tsi_type <> 'XXX' then
                chlorophyll and phosphorus need to accumulate and average
                values taken on the same date,
                secchi does not
                if v_hold_tsi_type in ('CHLOROPHYLL','PHOSPHORUS') and
                v_date_cnt > 0 then
                v_tsi_sum := v_tsi_sum + f_compute_tsi(v_hold_tsi_type,
                v_date_sum / v_date_cnt);
                v_tsi_cnt := v_tsi_cnt + 1;
                end if;
                p_end_tsi_type(v_hold_tsi_type, v_tsi_sum, v_tsi_cnt,
                v_tsi_rec);
          end if;
          v_hold_tsi_type := cv_res.tsi_type;
          v_tsi_sum := 0;
          v_tsi_cnt := 0;
          v_hold_date := to_date('01011900','mmddyyyy');
          end if;

          if v_hold_date <> cv_res.result_date then
          if v_hold_date <> to_date('01011900','mmddyyyy') then
```

```
      chlorophyll and phosphorus need to accumulate and average
      values taken on the same date,
      secchi does not
      if v_hold_tsi_type in ('CHLOROPHYLL','PHOSPHORUS') and
      v_date_cnt > 0 then
      v_tsi_sum := v_tsi_sum + f_compute_tsi(v_hold_tsi_type,
      v_date_sum / v_date_cnt);
      v_tsi_cnt := v_tsi_cnt + 1;
      end if;
end if;
v_date_sum := 0;
v_date_cnt := 0;
v_hold_date := cv_res.result_date;
end if;

if cv_res.tsi_type in ('SECCHI','SATELLITE') then
      convert all secchi readings to meters
case upper(cv_res.result_units_text)
      when 'M' then v_result_amt := cv_res.result_amt;
      when 'METERS' then v_result_amt := cv_res.result_amt;
      when 'FEET' then v_result_amt := cv_res.result_amt * 0.3048;
      when 'FT' then v_result_amt := cv_res.result_amt * 0.3048;
      when 'INCHES' then v_result_amt := cv_res.result_amt * 0.0254;
      when 'IN' then v_result_amt := cv_res.result_amt * 0.0254;
      when 'I' then v_result_amt := cv_res.result_amt * 0.0254;
      else /* assume feet */ v_result_amt := cv_res.result_amt *
      0.3048;
end case;
      check if the secchi hit bottom
select count(*) into v_temp_cnt
      from wt_swims_result_fact_v
      where fieldwork_seq_no = cv_res.fieldwork_seq_no and
            ((dnr_parameter_type = 'DNR_STORET' and
            dnr_parameter_code = 99420) or
            (dnr_parameter_type = 'SWIMS' and dnr_parameter_code =
            90943) or  -- added the two SWIMS parameters, 2/10/2010,
            b tinberg
            (dnr_parameter_type = 'SWIMS' and dnr_parameter_code =
            90942)) and
            result_value_no in ('Y','YES');
if v_temp_cnt > 0 then
      if cv_res.tsi_type = 'SECCHI' then
      v_tsi_rec.secchi_hit_bottom_flag := 'Y';
      elsif cv_res.tsi_type = 'SATELLITE' then
      v_tsi_rec.sat_secchi_hit_bottom_flag := 'Y';
      end if;
end if;
      check for stained water, first for Water column = CLEAR,
      second for color = BROWN, RED, or YELLOW
select count(*) into v_temp_cnt
      from wt_swims_result_fact_v
      where fieldwork_seq_no = cv_res.fieldwork_seq_no and
            dnr_parameter_type = 'SWIMS' and
            dnr_parameter_code = 90000 and
            result_value_no = 'CLEAR';
if v_temp_cnt > 0 then
      select count(*) into v_temp_cnt
```

```
            from wt_swims_result_fact_v
            where fieldwork_seq_no = cv_res.fieldwork_seq_no and
                  dnr_parameter_type = 'SWIMS' and
                  dnr_parameter_code = 90001 and
                  result_value_no in ('BROWN','RED','YELLOW');
            if v_temp_cnt > 0 then
            v_tsi_rec.secchi_stained_water_flag := 'Y';
            end if;
      end if;
      else
      if cv_res.result_value_no like '%ND%' or
      cv_res.result_qualifier_code = '2' then  -- below LOD, use 1/2 LOD
            v_result_amt := cv_res.lod_amt * 0.5;
            if cv_res.tsi_type = 'CHLOROPHYLL' then
            v_tsi_rec.chlor_below_lod_flag := 'Y';
        else
          v_tsi_rec.phos_below_lod_flag := 'Y';
        end if;
      else
        v_result_amt := cv_res.result_amt;
      end if;
      -- if mg/l units, convert to ug/L (that's what the TSI calculation
will expect)
      if upper(cv_res.result_units_text) like 'MG/L%' then
        v_result_amt := 1000 * v_result_amt;
      end if;
    end if;

    -- chlorophyll and phosphorus need to accumulate and average values
taken on the same date,
    -- secchi does not
    if cv_res.tsi_type in ('CHLOROPHYLL','PHOSPHORUS') then
      v_depth_ok := f_check_chlor_phos_depth(cv_res.sample_result_seq_no,
cv_res.sample_header_seq_no, cv_res.fieldwork_seq_no);

      if v_depth_ok then
        v_date_sum := v_date_sum + v_result_amt;
        v_date_cnt := v_date_cnt + 1;
      end if;
    else
      v_tsi_sum := v_tsi_sum + f_compute_tsi(cv_res.tsi_type,
v_result_amt);
      v_tsi_cnt := v_tsi_cnt + 1;
    end if;
  end loop;

  -- process last set of data
  if v_hold_station <> -1 then
    -- chlorophyll and phosphorus need to accumulate and average values
taken on the same date,
    -- secchi does not
    if v_hold_tsi_type in ('CHLOROPHYLL','PHOSPHORUS') and v_date_cnt > 0
then
      v_tsi_sum := v_tsi_sum + f_compute_tsi(v_hold_tsi_type, v_date_sum /
v_date_cnt);
      v_tsi_cnt := v_tsi_cnt + 1;
    end if;
    p_end_tsi_type(v_hold_tsi_type, v_tsi_sum, v_tsi_cnt, v_tsi_rec);
    p_end_station(v_tsi_rec);
    v_insert_cnt := v_insert_cnt + 1;
  end if;

  pk_swims_job_log.p_logit('Inserted ' || v_insert_cnt || ' records into
wt_swims_tsi_station_year.');
  v_dummy := pk_swims_job_log.write(true);
exception
  when others then
    rollback;
    pk_swims_job_log.p_logit('Exception raised in p_load_station_year,
v_hold_station = ' || v_hold_station || ': ' || sqlerrm);
    pk_swims_job_log.p_logit('Inserted ' || v_insert_cnt || ' records into
wt_swims_tsi_station_year.');
    v_dummy := pk_swims_job_log.write(true);
end;

procedure p_all_station_year_refresh(i_start_year in integer) is
begin
  delete from wt_swims_tsi_station_year;
  commit;

  for v_year in i_start_year..i_start_year+4 loop
  --for cv_year in (select distinct to_char(result_date_time,'yyyy')
res_year
  --                from wt_swims_result_fact_v) loop
    --p_load_station_year(to_number(cv_year.res_year));
    p_load_station_year(v_year);
  end loop;
end;

procedure p_load_wbic_year(i_year in number, i_wbic in number) is
  v_tsi_chlor          number;
  v_tsi_phos           number;
  v_tsi_secchi         number;
  v_tsi_sat_secchi     number;
  v_tot_tsi_chlor      number;
  v_tot_tsi_phos       number;
  v_tot_tsi_secchi     number;
  v_tot_tsi_sat_secchi number;
  v_tsi_chlor_cnt      integer;
  v_tsi_phos_cnt       integer;
  v_tsi_secchi_cnt     integer;
  v_tsi_sat_secchi_cnt integer;
  v_chlor_below_lod    varchar2(1);
  v_phos_below_lod     varchar2(1);
  v_secchi_hit_bottom  varchar2(1);
  v_stained_water      varchar2(1);
  v_sat_secchi_hit_bottom  varchar2(1);
begin
  delete from wt_swims_tsi_wbic_year
    where wbic = i_wbic and tsi_year = i_year;

  select sum(tsi.tsi_chlor_score_amt * tsi.tsi_chlor_samp_count),
sum(nvl(tsi.tsi_chlor_samp_count,0)),
         max(chlor_below_lod_flag), sum(tsi.tsi_secchi_score_amt *
tsi.tsi_secchi_samp_count),
         sum(nvl(tsi.tsi_secchi_samp_count,0)),
max(secchi_hit_bottom_flag),
         max(secchi_stained_water_flag), sum(tsi.tsi_phos_score_amt *
tsi.tsi_phos_samp_count),
         sum(nvl(tsi.tsi_phos_samp_count,0)),
         max(phos_below_lod_flag)
  into v_tot_tsi_chlor, v_tsi_chlor_cnt, v_chlor_below_lod,
v_tot_tsi_secchi, v_tsi_secchi_cnt,
       v_secchi_hit_bottom, v_stained_water, v_tot_tsi_phos,
v_tsi_phos_cnt, v_phos_below_lod
  from wt_swims_tsi_station_year tsi, wt_swims_monit_station ms,
wt_swims_monit_sta_isect_gv msi
  where tsi.monit_station_seq_no = ms.monit_station_seq_no and
        ms.monit_station_seq_no = msi.monit_station_seq_no and
        ms.station_type_code in ('LAKE','RESERVOIR','RIVERINE
IMPOUNDMENT') and
        ms.secondary_station_type = 'DEEPEST SPOT' and
        msi.intersection_code = 'WBODY' and
        msi.intersection_key = to_char(i_wbic) and
        tsi.tsi_year = i_year;

  select sum(tsi.tsi_sat_secchi_score_amt *
tsi.tsi_sat_secchi_samp_count),
         sum(nvl(tsi.tsi_sat_secchi_samp_count,0)),
max(sat_secchi_hit_bottom_flag)
  into v_tot_tsi_sat_secchi, v_tsi_sat_secchi_cnt, v_sat_secchi_hit_bottom
  from wt_swims_tsi_station_year tsi, wt_swims_monit_station ms,
wt_swims_monit_sta_isect_gv msi
  where tsi.monit_station_seq_no = ms.monit_station_seq_no and
        ms.monit_station_seq_no = msi.monit_station_seq_no and
        ms.station_type_code in ('LAKE','RESERVOIR','RIVERINE
IMPOUNDMENT') and
        --ms.secondary_station_type = 'DEEPEST SPOT' and
        msi.intersection_code = 'WBODY' and
        msi.intersection_key = to_char(i_wbic) and
        tsi.tsi_year = i_year;

    -- two samples of a parameter are needed to use the score
    --
    -- except satellite secchi which just needs one sample, per Jennifer
Filbert, 7/20/2009
    if v_tsi_chlor_cnt >= 2 or
       v_tsi_phos_cnt >= 2 or
       v_tsi_secchi_cnt >= 2 or
       v_tsi_sat_secchi_cnt >= 1 then
      if v_tsi_chlor_cnt >= 2 then
        v_tsi_chlor := v_tot_tsi_chlor / v_tsi_chlor_cnt;
      else
        v_tsi_chlor := null;
      end if;
      if v_tsi_phos_cnt >= 2 then
        v_tsi_phos := v_tot_tsi_phos / v_tsi_phos_cnt;
      else
        v_tsi_phos := null;
      end if;
      if v_tsi_secchi_cnt >= 2 then
        v_tsi_secchi := v_tot_tsi_secchi / v_tsi_secchi_cnt;
      else
        v_tsi_secchi := null;
      end if;
      if v_tsi_sat_secchi_cnt >= 1 then
        v_tsi_sat_secchi := v_tot_tsi_sat_secchi / v_tsi_sat_secchi_cnt;
      else
        v_tsi_sat_secchi := null;
      end if;

      insert into wt_swims_tsi_wbic_year
        (wbic, tsi_year, tsi_chlor_samp_count, tsi_phos_samp_count,
tsi_secchi_samp_count, tsi_sat_secchi_samp_count,
         tsi_chlor_score_amt, tsi_phos_score_amt, tsi_secchi_score_amt,
tsi_sat_secchi_score_amt,
         secchi_hit_bottom_flag, secchi_stained_water_flag,
sat_secchi_hit_bottom_flag, chlor_below_lod_flag, phos_below_lod_flag)
        values
        (i_wbic, i_year, v_tsi_chlor_cnt, v_tsi_phos_cnt,
v_tsi_secchi_cnt, v_tsi_sat_secchi_cnt,
         v_tsi_chlor, v_tsi_phos, v_tsi_secchi, v_tsi_sat_secchi,
         nvl(v_secchi_hit_bottom,'N'), nvl(v_stained_water,'N'),
nvl(v_sat_secchi_hit_bottom,'N'), nvl(v_chlor_below_lod,'N'),
nvl(v_phos_below_lod,'N'));
    end if;
end;

procedure p_all_wbic_year_refresh is
begin
  delete from wt_swims_tsi_wbic_year;

  for cv in (select distinct intersection_key, tsi_year
             from wt_swims_tsi_station_year tsi, wt_swims_monit_station
ms, wt_swims_monit_sta_isect_gv msi
             where tsi.monit_station_seq_no = ms.monit_station_seq_no and
                   ms.monit_station_seq_no = msi.monit_station_seq_no and
                   ms.station_type_code in ('LAKE','RESERVOIR','RIVERINE
IMPOUNDMENT') and
                   msi.intersection_code = 'WBODY') loop
    p_load_wbic_year(cv.tsi_year, to_number(cv.intersection_key));
  end loop;
end;

procedure p_load_wbic(i_wbic in varchar2) is
  v_year_cnt       integer;
  v_samp_cnt       integer;
  v_tsi_score      wt_swims_tsi_wbic.tsi_score_amt%type;
  v_tsi_type       wt_swims_tsi_wbic.tsi_type_code%type;
  v_below_lod      wt_swims_tsi_wbic.below_lod_flag%type;
  v_hit_bottom     wt_swims_tsi_wbic.secchi_hit_bottom_flag%type;
  v_stained_water  wt_swims_tsi_wbic.secchi_stained_water_flag%type;
  v_au_seq_no      wt_assessment_unit.assessment_unit_seq_no%type;
begin
  delete from wt_swims_tsi_wbic where wbic = i_wbic;
  -- see if we can find an exact match for an assessment unit
  begin
```

```
    select assessment_unit_seq_no into v_au_seq_no
      from wt_assessment_unit
      where wbic = i_wbic and
            water_type not like '%BEACH%';
  exception
    when NO_DATA_FOUND or TOO_MANY_ROWS then
      -- try lookup table
      begin
        select assessment_unit_seq_no into v_au_seq_no
          from wt_swims_tsi_wbic_au_lookup
          where wbic = i_wbic;
      exception
        when NO_DATA_FOUND then
          v_au_seq_no := null;
      end;
  end;

  -- check for three years of chlorophyll
  select count(distinct tsi.tsi_year), avg(tsi.tsi_chlor_score_amt),
sum(tsi.tsi_chlor_samp_count),
         max(chlor_below_lod_flag)
    into v_year_cnt, v_tsi_score, v_samp_cnt, v_below_lod
    from wt_swims_tsi_wbic_year tsi
    where tsi.tsi_chlor_score_amt > 0 and
          tsi.wbic = i_wbic;
  if v_year_cnt >= 3 then
    insert into wt_swims_tsi_wbic
      (wbic, tsi_score_amt, tsi_type_code, sample_count, year_count,
below_lod_flag, assessment_unit_seq_no)
      values
      (i_wbic, v_tsi_score, 'CHLOROPHYLL', v_samp_cnt, v_year_cnt,
v_below_lod, v_au_seq_no);
  else
    -- check for three years of secchi
    select count(distinct tsi.tsi_year), avg(tsi.tsi_secchi_score_amt),
sum(tsi.tsi_secchi_samp_count),
           max(secchi_hit_bottom_flag), max(secchi_stained_water_flag)
      into v_year_cnt, v_tsi_score, v_samp_cnt, v_hit_bottom,
v_stained_water
      from wt_swims_tsi_wbic_year tsi
      where tsi.tsi_secchi_score_amt > 0 and
            tsi.wbic = i_wbic;
    if v_year_cnt >= 3 then
      insert into wt_swims_tsi_wbic
        (wbic, tsi_score_amt, tsi_type_code, sample_count, year_count,
secchi_hit_bottom_flag, secchi_stained_water_flag, assessment_unit_seq_no)
        values
        (i_wbic, v_tsi_score, 'SECCHI', v_samp_cnt, v_year_cnt,
v_hit_bottom, v_stained_water, v_au_seq_no);
    else
      -- check for three years of satellite secchi
      select count(distinct tsi.tsi_year),
avg(tsi.tsi_sat_secchi_score_amt), sum(tsi.tsi_sat_secchi_samp_count),
             max(sat_secchi_hit_bottom_flag)
        into v_year_cnt, v_tsi_score, v_samp_cnt, v_hit_bottom
        from wt_swims_tsi_wbic_year tsi
        where tsi.tsi_sat_secchi_score_amt > 0 and
```

```
              tsi.wbic = i_wbic;
      if v_year_cnt >= 3 then
        insert into wt_swims_tsi_wbic
          (wbic, tsi_score_amt, tsi_type_code, sample_count, year_count,
secchi_hit_bottom_flag, assessment_unit_seq_no)
          values
          (i_wbic, v_tsi_score, 'SATELLITE', v_samp_cnt, v_year_cnt,
v_hit_bottom, v_au_seq_no);
      end if;
    end if;
  end if;
end;

procedure p_all_wbic_refresh is
begin
  delete from wt_swims_tsi_wbic;

  for cv_wbic in (select distinct wbic from wt_swims_tsi_wbic_year tsi)
loop
    p_load_wbic(cv_wbic.wbic);
  end loop;
end;
•
procedure p_complete_refresh(i_start_year in integer) is
begin
  p_all_station_year_refresh(i_start_year);
  p_all_wbic_year_refresh;
  p_all_wbic_refresh;
end;

end;
/
```