Database documentation: plankton

K. A. Mackay

NIWA Internal Report No. 69
2000

Contents

List of Figures

1 Database documentation series

The National Institute of Water and Atmospheric Research (NIWA) currently carries out the role of Data Manager and Custodian for the fisheries research data owned by the Ministry of Fisheries (MFish).

The Ministry of Fisheries data set incorporates historic research data, data collected more recently by MAF Fisheries prior to the split in 1995 of Policy to the Ministry of Fisheries and research to NIWA, and currently data collected by NIWA and other agencies for the Ministry of Fisheries.

This document provides an introduction to the plankton survey database **plankton,** and is a part of the database documentation series produced by NIWA.

All documents in this series include an introduction to the database design, a description of the main data structures accompanied by an Entity Relationship Diagram (ERD), and a listing of all the main tables. The ERD graphically shows how all the tables fit in together, and their relationships to other databases.

This document is intended as a guide for users and administrators of the **plankton** database.

2 Marine Research Plankton Survey Database

Plankton surveys and primarily, egg surveys, are another tool to help fisheries scientists calculate biomass abundance[1]. The basic theory of population estimates made from egg surveys is well known:

> "*If one can estimate the total production of eggs or larvae of a stock,* ***P****, throughout a spawning season, and determine the mean fecundity,* ***F****, of a mature female and the proportion of females in a mature stock,* ***K****, then the total abundance of the mature stock*
>
> $$M = \frac{P}{KF}$$ "[2]

Plankton surveys in a sample area can be conducted in a variety of ways such as stratified, random, or grid stations and are often carried out in conjunction with trawl surveys. The nets used are varied in size and shape but all have very fine mesh They range from a large

[1] Zeldis, J. 1993. The applicability of egg surveys for spawning stock biomass estimation of snapper, orange roughy and hoki in New Zealand. *Bulletin of Marine Science* 53(2) : 864-890

[2] Crossland, J. 1980. The number of snapper *Chrysophrys auratus* (Forster), in the Hauraki Gulf, New Zealand, based on egg surveys in 1974-75 and 1975-76. *Fisheries Research Bulletin No. 22* 40p.

Engels midwater trawl for rock lobster pureluii, to small cylinder-code trawl nets, to vertically dropped plankton nets.

These surveys use a variety of egg production models (such as the: “daily egg production method” (DEPM); the “annual egg production method” (AEPM); and the “daily fecundity reduction method” (DFRM)) for fish stock biomass calculation.

The DEPM estimates the adult spawning biomass from the ratio of the daily production of planktonic eggs and the daily fecundity. The latter is calculated using daily spawning frequency, average batch fecundity, average female weight and sex ratio. The planktonic eggs are caught during a plankton survey and the spawning frequency, batch fecundity, and sex ratio are estimated using a trawl survey.

The DFRM estimates biomass of spawning females by dividing the daily planktonic egg production in the survey area by the weight-specific daily fecundity of females. Because it is a daily method, there is the advantage that the DFRM does not need to cover the entire spawning season.

3 Data Structures

Initially, plankton surveys were included as part of the **trawl** database. However, it soon became apparent that the **trawl** data structures were not suitable for plankton catch data. Therefore, a separate database, **plankton**, was created that, at the conceptual level, is very similar to **trawl**[3], but has attributes specific to plankton catches. This has the advantage of allowing easy access to related trawl survey data.

3.1 Table Relationships

This database contains several tables. The ERD for **plankton** (Figure 1) shows the physical data model structure[4] of the database and its entities (each entity is implemented as a database *table*) and relationships between these tables. Each table represents an object, event, or concept in the real world that has been represented in the database. Each *attribute* of a table is a defining property or quality of the table.

All of the table’s attributes are shown in the ERD. The underlined attributes represent the table’s primary key[5]. This schema is valid regardless of the database system chosen, and it can remain correct even if the Database Management System (DBMS) is changed.

[3] Mackay, K. 1998: Marine Research database documentation. 6. trawl. *NIWA Greta Point Internal Report No. 6.* 35p.
[4] Also known as a database *schema*.
[5] A primary key is an attribute or a combination of attributes that contains an unqiue value to identify that record.

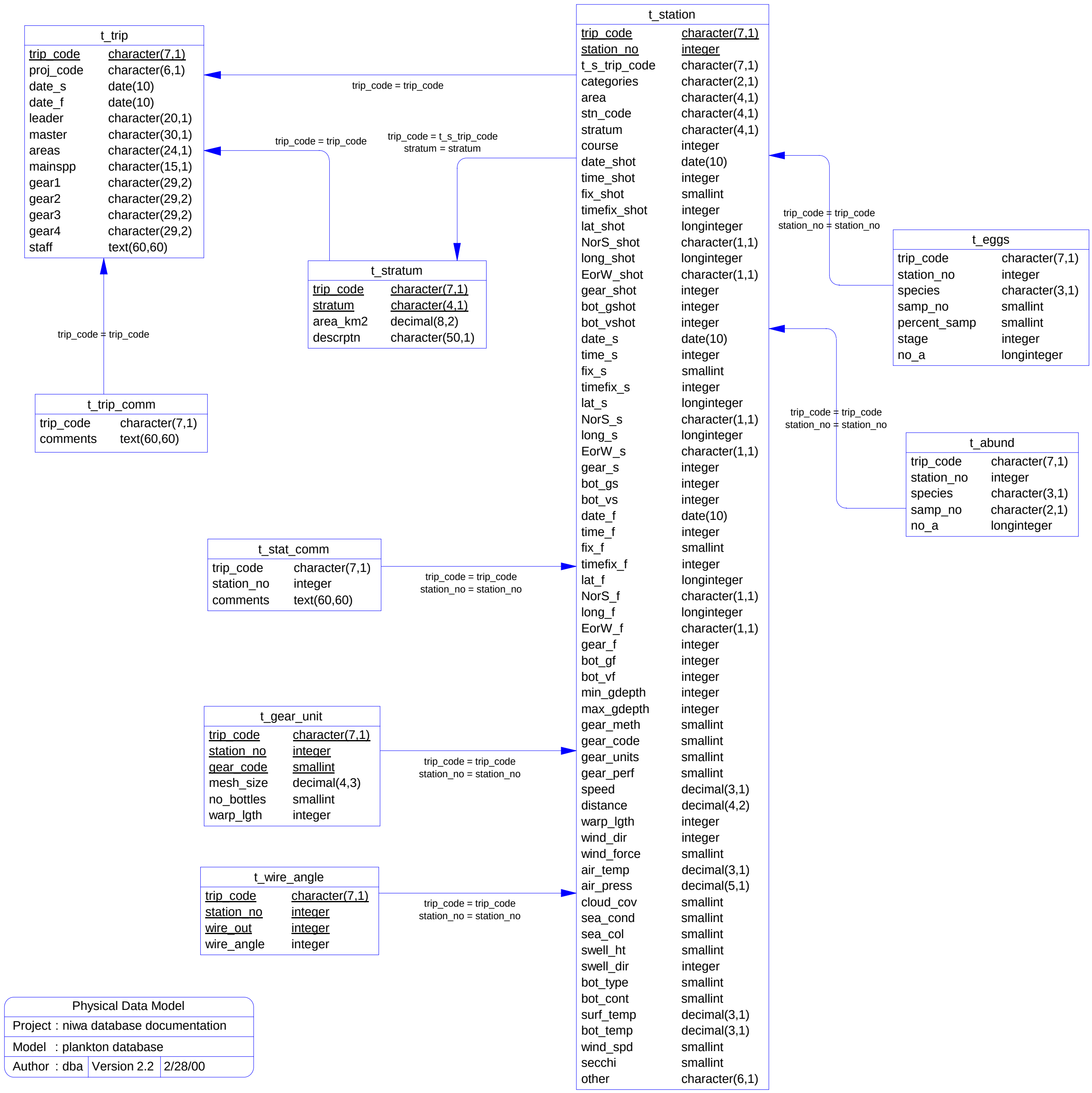

Figure 1: Entity Relationship Diagram (ERD) for the plankton database.

Note that Figure 1 shows the main tables only. Some of the tables in the **plankton** database have attributes, called foreign keys[6], which contain standard NIWA fisheries codes, such as *species*. These attributes provide links to tables in **plankton** and the **rdb** (research database) database, which contains the definitive list of standard codes. An expanded ERD for these tables is shown in Figure 2.

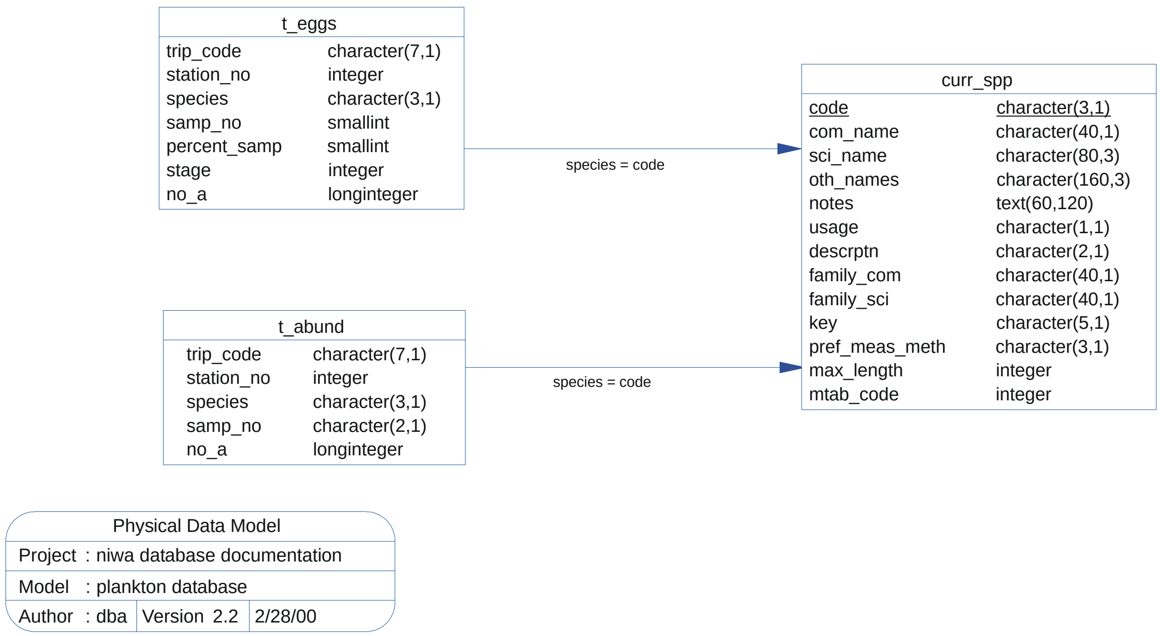

Figure 2: Expanded ERD of *t_eggs* and *t_abund* showing relationships to the *curr_spp* table in the rdb database.

Section 5 shows a listing of all the **plankton** tables as implemented by the Empress DBMS. As can be seen in the listing of the tables, a table's primary key has an unique index on it. Primary keys are generally listed using the format:

Indices: `UNIQUE index_name ON (`*`attribute`*` [, `*`attributes`*` ])`

where the attribute(s) make up the primary key and the index name is the primary key name. Note that the typographical convention for the above (and subsequent) format is the square brackets [ ] may contain an item that is repeated zero or more times.

[6] A foreign key is any attribute, or a combination of attributes, in a table that is a primary key of another table. Tables are linked together through foreign keys.

This unique index prevents records with duplicate key values from being inserted into the table, e.g., a new trip with an existing trip code, and hence ensures that every record can be uniquely identified.

The **plankton** database is implemented as a relational database. That is, each table is a special case of a mathematical construct known as a *relation* and hence elementary relation theory is used to deal with the data within tables and their relationships between them. All relationships in **plankton** are of the type *one-to-many*[7]. This is shown in the ERD by connecting a single line (indicating 'many') from the child table (e.g., *t_station*) to the parent table (e.g., *t_trip*) with an arrowhead (indicating 'one') pointing to the parent.

Every relationship has a mandatory or optional aspect to it. That is, if a relationship is mandatory, then it has to occur and least once, while an optional relationship might not occur at all. For example, in Figure 1, consider that relationship between the table *t_trip* and it's child table *t_trip_comm*. The symbol "O" by the child *t_trip_comm* means that a fish can have zero or many trip comments, while the bar by the parent *t_trip* means that for every trip comment there must be a matching trip.

Most of the tables in **plankton** contain foreign keys, which link these tables to each other and to tables in the **rdb** database (Figure 1 and Figure 2). The majority of these links are enforced by referential constraints[8]. Constraints do not allow *orphans* to exist in any table, i.e., where a child record exists without a related parent record. This may happen when: a parent record is deleted; the parent record is altered so that the relationship is lost; or a child record is entered without a parent record. Constraints are shown in the table listings by the following format:

```
Referential:      error message (attribute[, attribute]) INSERT
                  parent table (attribute[, attribute])
```

For example, consider the following constraint found in the table *t_station*:

```
Referential:      invalid trip code (trip_code) INSERT
                  t_trip (trip_code)
```

This means that the value of the attribute *trip_code* of a record upon insert into *t_station* must already exist in the parent table *t_trip* or the record will be rejected and an error message will be displayed.

[7] A one-to-many relationship is where one record in a table (the *parent*) relates to one or many records in another table (the *child*).
[8] Also known as integrity checks.

All tables in this database are indexed. That is, attributes that are most likely to be used as a searching key have like values linked together to speed up searches. These indices are listed using the following format:

`Indices:` `NORMAL (2, 15) index_name ON (`*`attribute`*`[, `*`attribute`*`])`

Note that indices may be simple, pointing to one attribute or composite pointing to more than one attribute. The numbers "…(2, 15)…" in the syntax are Empress DBMS default values relating to the amount of space allocated for the index.

3.2 Database design

As reflected by the ERD, the highest level of a plankton survey is a research trip. Details for each trip are held in the table *t_trip* (Table 1). Each trip is uniquely identified by a trip code, stored as the attribute *trip_code.*

Note that the comments for a trip are held in a separate table *t_trip_comm* (Table 2). This means that one trip may have zero, one, or more than one comment associated with it. It can be argued that there is a one-to-one relationship between *t_trip* and *t_trip_comm*. After all, all comments to be made about one trip can be made in one comment. However, the trip comments have been separated from the trip details to two reasons:

I. Comments can be recorded at any time during a trip life-cycle. Rather than adding comments to those already recorded, it is easier just to create a new record.
II. To optimise query times, attributes with long field sizes, such as comments, are placed in separate tables to avoid being hit during tables scans for a regular expression.

For stratified plankton surveys, stratum details, such as stratum code and area (in square kilometres) are stored in the table *t_stratum* (Table 3). These strata may not necessarily be the same as those used for trawling during the same trip (as recorded in the **trawl** database). Notice that there is an optional link from *t_trip* to *t_stratum*; this means that not all trips have to have strata, i.e., unstratified plankton surveys.

Any one trip also relates to many stations. This is a mandatory relationship: a trip has to have at least one station before it can be entered into the database. Generally, a station is the location at which the plankton gear was deployed. Details for the station, such as shot start and finish location, time, depth, gear performance and environment parameters are stored in the table *t_station* (Table 4).

This table differs from the standard station table in the **trawl** database because it allows for up to 3 positions to be recorded (shot, start, and finish) for a station: the shot time and position is taken where the gear is first deployed; the start time and position is taken where the gear has reached the target depth and has started to be hauled; and the finish time and position is taken when the net reaches the sea surface. This is illustrated in Figure 3.

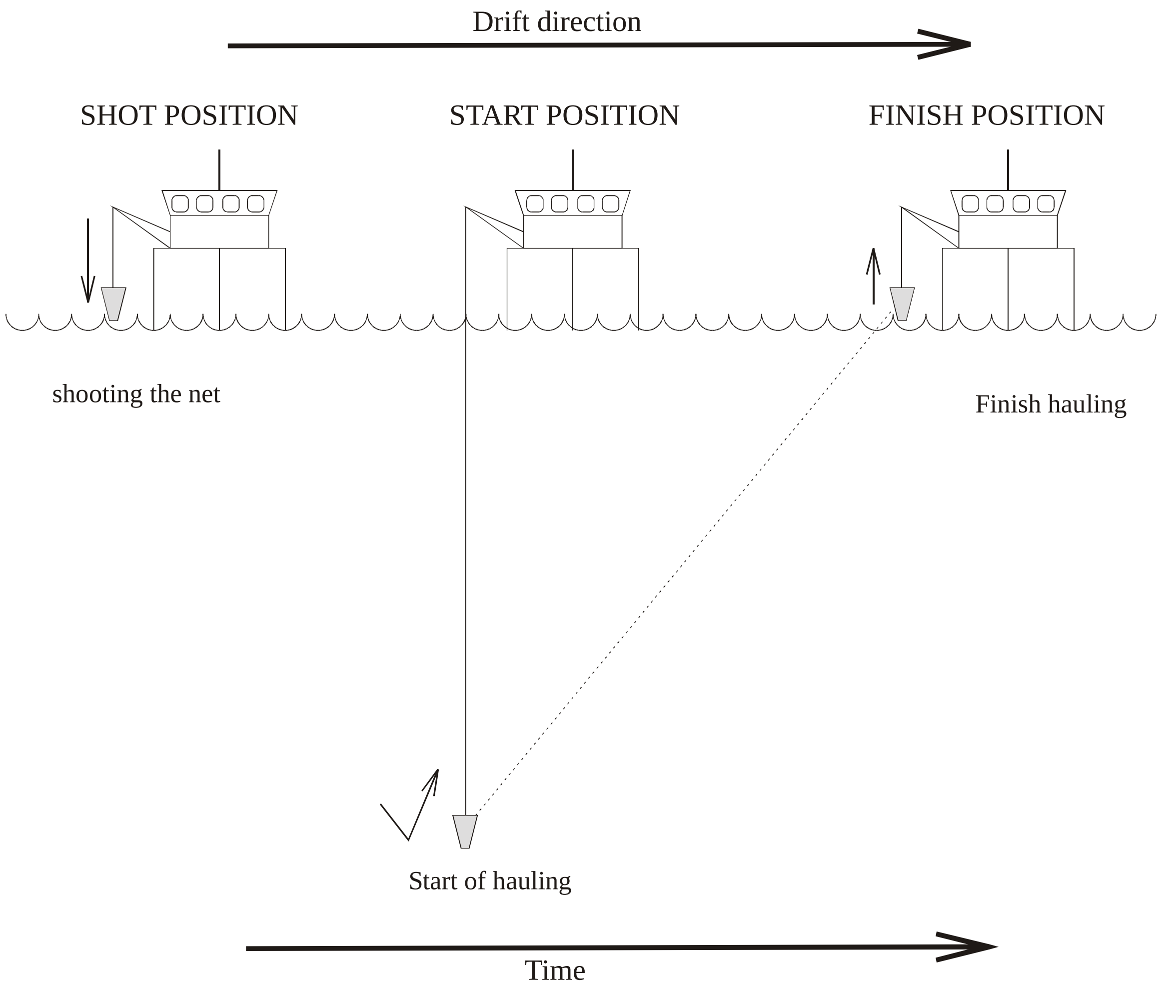

Figure 3: Origin of the three sets of time and position for a station.

In some instances, the shot and start positions of the gear are one in the same thing. However, usually a gear does not start to collect samples until is has reached it's target depth and opened by a remote trigger, or when it get forward motion from when then wire has stopped being paid out and the vessel moves forward and/or the wire has started to be hauled in.

Note that a station may or may not occur within a pre-defined stratum (the table *t_station* contains the attribute *stratum*) and that one stratum may or may not contain stations. Therefore, there is a two-way optional many-to-one relationship between *t_station* and *t_stratum*.

Like the table *t_trip*, *t_station* has its own comments table *t_stat_comm* (Table 5). The same arguments that have been used for the creation of the *t_trip_comm* table also apply here.

For plankton stations prior to the introduction of net sondes and depth-meters, the depth and path of the plankton gear was estimated from the angle of the gear wires at certain intervals or wire length. These wire angle data are stored in the table *t_wire_angle* (Table 6), and are linked to *t_station* by the foreign keys *trip_code* and *station_no*.

Several types of plankton stations involve multiple units of gear, each with their own mesh size and wire winches. In such cases, details of these gears are recorded in the table *t_gear_unit* (Table 7), and are linked to *t_station* by the foreign keys *trip_code* and *station_no.* The default for a single gear unit is for all data to be recorded in *t_station*.

Samples of fish eggs are taken from each tow and are staged under a microscope (by the number of cell divisions that have taken place). The table *t_eggs* (Table 8) records the counts of the number of eggs for each stage as collected for each sample from each catch.

Samples of the total catch from the plankton tow are also sampled to determine the relative abundance of species. These abundance data are recorded in the table *t_abund* (Table 9). This includes abundance of fish, salps, zooplankton and microzooplankton.

4 Table Summaries

This database has nine tables containing information pertaining to research plankton surveys. The following is a listing of the tables contained in the **plankton** database:

1. **t_trip :** contains profile information on all trips.
2. **t_trip_comm :** contains comments for a particular trip.
3. **t_stratum :** contains details of strata surveyed for a trip.
4. **t_station :** contains data on location, gear used and environment at each station within a trip.
5. **t_stat_comm :** contains comments for a station in a trip.
6. **t_wire_angle :** contains wire angles (from the vertical) during gear retrieval.
7. **t_gear_units :** contains mesh sizes, warp lengths and other details when more than one unit of gear is used during a station.
8. **t_eggs :** contains stage frequency data for samples of eggs caught at each station on a trip.
9. **t_abund :** contains total abundance of different species from samples taken from the catch of a plankton net.

5 plankton Tables

The following are listings of the tables in the **plankton** database, including attribute names, data types (and any range restrictions), and comments.

See Appendix 1 for attributes that have comments referring to the Trawl Instructions (unpub. NIWA report).

5.1 Table 1: t_trip

Comment: Profile information on all trips held in this database.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1) smatch “[a-z0-9][a-z0-9][a-z0-9][6-9][0-9][0-3][0-9]”	No	Trip code - 3 char vessel name, 2 digit year and 2 digit trip number
proj_code	character(6,1) smatch “[A-Z][A-Z][A-Z][A-Z][0-9][0-9]”		Project or programme code for this trip as in the management database
date_s	date(5)		Start date for the trip.
date_f	date(5)		Finish date for the trip
leader	character(20,1)		Name of trip leader
master	character(30,1)		Name of trip master(s)
areas	character(24,1)		Codes of area(s) surveyed separated by commas (,)
mainspp	character(15,1) smatch “{[A-Z,]}”		Target species code(s) separated by commas
gear1	character(29,2) match “{[0-9,.]}”		Codend, liner & cover mesh sizes (mm), ground rope, sweep & bridle lengths (m) separated by commas for 1st gear code used
gear2	character(29,2) match “{[0-9,.]}”		Codend, liner & cover mesh sizes (mm), ground rope, sweep & bridle lengths (m) separated by commas for 2nd gear code used

Attributes	Data Type	Null?	Comment
gear3	character(29,2) match “{[0-9,.]}”		Codend, liner & cover mesh sizes (mm), ground rope, sweep & bridle lengths (m) separated by commas for 3rd gear code used
gear4	character(29,2) match “{[0-9,.]}”		Codend, liner & cover mesh sizes (mm), ground rope, sweep & bridle lengths (m) separated by commas for 4th gear code used
staff	text(20,60,20,1)		Name(s) of all staff on the trip

Creator: dba
Indices: UNIQUE trip_key BTREE ON (trip_code)

5.2 Table 2: t_trip_comm

Comment: Comments for a particular trip.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
comments	text(60,120,60,1)		Any comments about this trip; e.g., details about gear used apart from those recorded in the trip table

Creator: dba
Referential: invalid trip_code (trip_code) INSERT t_trip (trip_code)
Indices: NORMAL (2, 15) BTREE ON (trip_code)

5.3 Table 3: t_stratum

Comment: Table of strata surveyed in all trips.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
stratum	character(4,1)	No	Stratum code - unique within a trip
area_km2	decimal(8,2) > 0.00		Size of a stratum in square kilometres (km^2) - must be greater than 0 km^2.
descrptn	character(50,1)		Short description of the stratum; e.g., location, depth ranges

Creator: dba
Referential: invalid trip_code (trip_code) INSERT t_trip (trip_code)
Indices: NORMAL (2, 15) BTREE ON (stratum)
NORMAL (2, 15) BTREE ON (area_km2)
UNIQUE stra_key BTREE ON (trip_code, stratum)

5.4 Table 4: t_station

Comment: Data on location, gear used and environmental conditions at each station during a trip.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
categories	character(2,1)		Two separate 1-character user defined categories; definitions should be in trip comments
area	character(4,1)		Code describing area, refer to rdb:area_codes
stn_code	character(4,1)		Code for a permanent station occupied repeatedly
stratum	character(4,1)		Stratum number if the trip was a stratified survey, else a transect code
course	integer range 0 - 359		Course of vessel during the shot (course-made-good)
date_shot	date(5)		Date when gear is first deployed (dd Mmm yy format)
time_shot	integer range 0 - 2359		Time (24hr,NZST) when gear is first deployed
fix_shot	character(2,1)		Method of fixing position at gear deployment. Refer to rdb:t_fix_meth_codes
timefix_shot	integer		Time (in minutes) elapsed since last position fix at gear deployment
lat_shot	longinteger match “[3-6][0-9][0-5][0-9][0-9][0-9]”		Latitude of vessel at gear deployment (ddmmmm format, d=deg., m=min. to 2 implied dec. pl.)
NorS_shot	character(1,1) smatch “[NS]”		Gear deployment position hemisphere
long_shot	longinteger match “1[7-8][0-9][0-5][0-9][0-9][0-9]”		Longitude of vessel at gear deployment (dddmmmm format, d=deg., m=min. to 2 implied dec. pl.)

Attributes	Data Type	Null?	Comment
EorW_shot	character(1,1) smatch “[EW]”		Gear deployment position meridian
bot_vshot	integer		Depth (m) of sea bottom at vessel position at deployment
date_s	date(5)		Starting date of the shot (dd Mmm yy format)
time_s	integer range 0 - 2359		Starting time (24hr,NZST) of the shot (hhmm format)
fix_s	character(2,1)		Method of fixing position at start of tow, refer rdb:t_fix_meth_codes.
timefix_s	integer		Time (in minutes) elapsed since last position fix at the start of tow
lat_s	longinteger match “[3-6][0-9][0-5][0-9][0-9][0-9]”		Latitude of vessel at start of tow (ddmmmm format, d=deg., m=min. to 2 implied dec. pl.)
NorS_s	character(1,1) smatch “[NS]”		Tow start position hemisphere
long_s	longinteger match “1[7-8][0-9][0-5][0-9][0-9][0-9]”		Longitude of vessel at start of tow (dddmmmm format, d=deg., m=min. to 2 implied dec. pl.)
EorW_s	character(1,1) smatch “[EW]”		Tow start position meridian
gear_s	integer		Depth (m) of gear at tow start
bot_gs	integer		Depth (m) of sea bottom at gear position at tow start
bot_vs	integer		Depth (m) of sea bottom at vessel position at tow start
date_f	date(5)		Finishing date of the shot (dd Mmm yy format)
time_f	integer range 0 - 2359		Finishing time (24hr,NZST) of the shot (hhmm format)
fix_f	character(2,1)		Method of fixing position at finish of tow, refer rdb:t_fix_meth_codes.

Attributes	Data Type	Null?	Comment
timefix_f	integer		Time (in minutes) elapsed since last position fix at the finish of tow
lat_f	longinteger match “[3-6][0-9][0-5][0-9][0-9][0-9]”		Latitude of vessel at finish of tow (ddmmmm format, d=deg., m=min. to 2 implied dec. pl.)
NorS_f	character(1,1) smatch “[NS]”		Tow finish position hemisphere
long_f	longinteger match “1[7-8][0-9][0-5][0-9][0-9][0-9]”		Longitude of vessel at finish of tow (dddmmmm format, d=deg., m=min. to 2 implied dec. pl.)
EorW_f	character(1,1) smatch “[EW]”		Tow finish position meridian
gear_f	integer		Depth (m) of gear at tow finish
bot_gf	integer		Depth (m) of sea bottom at gear position at tow finish
bot_vf	integer		Depth (m) of sea bottom at vessel position at tow finish
min_gdepth	integer		Min depth (m) of lowest part of gear during tow
max_gdepth	integer		Max depth (m) of lowest part of gear during tow
gear_meth	character(2,1)		Gear method code, descriptions in rdb:meth_codes
gear_code	smallint		Code for set of gear used, details in trip record. Also used as a link to t_gear_unit
gear_units	smallint		Number of units of gear used in the tow
gear_perf	smallint range 1 - 4		Code for performance of gear during the tow, refer to the trawl instructions
speed	decimal(3,1)		Average speed through water during shot (knots)
distance	decimal(4,2)		Distance of gear over bottom (nautical miles)
warp_lgth	integer		Length of warp during the tow (m)

Attributes	Data Type	Null?	Comment
wind_dir	integer range 0 - 359, = 999		Wind (true) direction; 999=No wind
wind_force	smallint range 0 - 12		Wind force on Beaufort scale
air_temp	decimal(3,1)		Air temperature (degrees C)
air_press	decimal(5,1)		Air pressure (millibars)
cloud_cov	smallint range 0 - 8		Code describing cloud cover during tow refer to trawl instructions
sea_cond	smallint range 0 - 9		Code describing condition of sea, refer trawl instructions
sea_col	smallint range 1 - 8		Code describing colour of sea, refer trawl instructions
swell_ht	smallint range 1 - 3		Code describing height of swell, refer trawl instructions
swell_dir	integer range 0 - 359, i= 999		Direction (true) of the swell
bot_type	smallint range 0 - 9		Code describing sea bottom type, refer trawl instructions
bot_cont	smallint range 0 - 5		Code describing sea bottom contour, refer trawl instructions
surf_temp	decimal(3,1)		Surface temperature (degrees C)
bot_temp	decimal(3,1)		Temperature at bottom (degrees C)
wind_spd	smallint		Wind speed (m/s) from anemometer (1kt=0.51m/s)
secchi	smallint		Depth when Secchi disc is invisible (m)
other	character(6,1)		Any other details, should be fully commented.

Creator: dba

Referential:
invalid trip_code (trip_code) INSERT t_trip (trip_code)
invalid area code (area) INSERT rdb : area_codes (code)
invalid fix_s code (fix_s) INSERT rdb :
      t_fix_meth_codes (fix_meth_code)
invalid fix_f code (fix_f) INSERT rdb :
      t_fix_meth_codes (fix_meth_code)
invalid gear code (gear_meth) INSERT rdb:meth_codes (code)

Indices:
UNIQUE BTREE stat_key ON (trip_code, station_no)
NORMAL (2, 15) BTREE ON (station_no)

5.5 Table 5: t_stat_comm

Comment: Comments for a station in a trip.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
comments	text(60,120,60,1)		Comments for this station - should include comments about catch data or any special action taken during tow

Creator: dba
Referential: invalid trip_code, station_no (trip_code, station_no)
INSERT t_station (trip_code, station_no)
Indices: NORMAL (2, 15) BTREE ON (trip_code)
NORMAL (2, 15) BTREE ON (station_no)

5.6 Table 6: t_wire_angle

Comment: Details of wire angles (from the vertical) of the gear wire during retrieval. Used prior to depth-meters to estimate the track of the gear during use.

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
wire_out	integer		Metres of wire out.
wire_angle	integer range 0 - 90		Angle (from the vertical) of the wire.

Creator: dba
Referential: invalid trip_code, station_no (trip_code, station_no)
INSERT t_station (trip_code, station_no)
Indices: UNIQUE BTREE wire_angle_pk ON (trip_code, station_no, wire_out)

5.7 Table 7: t_gear_unit

Comment: Details of an individual gear when stations have multiple gear units.

Attributes	**Data Type**	**Null?**	**Comment**
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
gear_code	smallint	No	Unique number to identify a gear unit within a station.
mesh_size	decimal(4,3)		Smallest mesh size (mm) of the gear.
no_bottles	smallint		Number of sample bottles attached to the gear.
warp_lgth	integer		Length of warp (m) for gear unit during tow.

Creator: dba
Referential: invalid trip_code, station_no (trip_code, station_no)
INSERT t_station (trip_code, station_no)
Indices: UNIQUE BTREE gear_unit_pk ON (trip_code, station_no,
gear_unit)

5.8 Table 8: t_eggs

Comment: Table for egg stages and abundance for each species in each station on a trip

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
species	character(3,1)	No	3-char species code, refer to rdb:curr_spp
samp_no	smallint		Sequential number to identify each sub-sample of a species taken from the whole catch for that species
percent_samp	smallint		Sampling percentage associated with this record
stage	integer		Egg development stage for this species
no_a	longinteger		Total count of eggs for this stage for this species

Creator: dba

Referential: invalid trip_code, station_no (trip_code, station_no) INSERT t_station (trip_code, station_no)
invalid species code (species) INSERT rdb:curr_spp (code)

Indices:
NORMAL (2, 15) BTREE ON (station_no)
NORMAL (2, 15) BTREE ON (species)
NORMAL (2, 15) BTREE ON (stage)
NORMAL (2, 15) BTREE ON (trip_code)

5.9 Table 9: t_abund

Comment: Table for abundance of species taken from samples of catches from plankton trawls

Attributes	Data Type	Null?	Comment
trip_code	character(7,1)	No	Trip code as defined in the trip table
station_no	integer	No	Station number - unique in a trip
species	character(3,1)	No	3-char species code, refer to rdb:curr_spp
samp_no	smallint		Sequential number to identify each sub-sample of a species taken from the whole catch for that species
no_a	longinteger		Total count of a species in a sample

Creator: dba
Referential: invalid trip_code, station_no (trip_code, station_no) INSERT t_station (trip_code, station_no)
invalid species code (species) INSERT rdb:curr_spp (code)
Indices: NORMAL (2, 15) BTREE ON (trip_code)
NORMAL (2, 15) BTREE ON (station_no)
NORMAL (2, 15) BTREE ON (species)

6 plankton Business Rules

6.1 Introduction to business rules

The following are a list of business rules pertaining to the age database. A business rule is a written statement specifying what the information system (i.e., any system that is designed to handle plankton data) must do or how it must be structured.

There are three recognised types of business rules:

Fact	Certainty or an existence in the information system
Formula	Calculation employed in the information system
Validation	Constraint on a value in the information system

Fact rules are shown on the ERD by the cardinality (e.g., one-to-many) of table relationships. Referential constraints, range checks, and algorithms both in the database and during data validation implement the formula and validation type rules.

6.2 Summary of rules

Trip details (t_trip)

trip_code	Trip code, must be unique. Trip codes are in the following format: 3 character vessel code (see the *vessels* table in the **rdb** database for available codes); 2 digit year (e.g., 99 = 1999, 00 = 2000); 2 digit sequential trip number for each vessel each year.
project_code	Project code must be a valid code within the NIWA project management system (unknown project code = "NULL00").
date_s	The start date of the trip must be a legitimate date.
date_f	The finish date of the trip must be a legitimate date.
	Multiple column check on trip dates The finish date must be later than or equal to the start date.
areas	Each of the listed area codes must be a valid area code as listed in the *area_codes* table in the **rdb** table.
mainspp	Each of the listed species codes must be a valid species code as listed in the *curr_spp* table in the **rdb** database.
gear1 – gear4	Gear descriptions. The following describe the format, and where applicable, the business rules for the description of gear used during a trip:

gear number	Must be a unique, sequential number from 1 to 6 to identify each gear.
gear method	Must be a valid code as listed in the *meth_codes* table in the **rdb** database.
codend mesh	Null, or a number greater than 0.
liner mesh	Null, or a number greater than 0.
cover mesh	Null, or a number greater than 0.
ground rope length	Null, or a number greater than 0.
ground rope height	Null, or a number greater than 0.

gear1 - gear4 (cont…)

sweep length	Null, or a number greater than 0.
bridle length	Null, or a number greater than 0.
default headline height	Null, or a number greater than 0.
headline height code	If the default headline height is not null, then must be a valid code as listed in the *t_headline_codes* table in the **rdb** database
default wing distance	Null, or a number greater than 0.
wing distance code	If the default wing distance is not null, then must be a valid code as listed in the *t_wing_dist_codes* table in the **rdb** database
default door distance	Null, or a number greater than 0.
door distance code	If the default door distance is not null, then must be a valid code as listed in the *t_door_dist_codes* table in the **rdb** database

Trip comments (t_trip_comm)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

Stratum details (t_stratum)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

area_km2 Must be a number greater than 0.

Station details (t_station)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

station_no Must be a unique number within a single trip.

area Area code must be a valid code as listed in the *area_codes* table in the **rdb** database.

course Course must be within the range of 0 – 359 degrees.

date_shot The date at the shooting of the gear at a station must be a legitimate date.

Multiple column checks on shot date:
The date must fall within the range of the range of the trip start and finish dates.

time_shot Shot time of the station must be a valid 24-hour time and fall within the range of 0 – 2359 hours.

fix_shot }
fix_s }
fix_f } The method of position fix code must be valid code as listed in the *t_fix_meth_codes* table in the **rdb** database. Also listed in Appendix 1.

lat_shot Must be a valid latitude

NorS_shot Northern or Southern Hemisphere at the shooting of the gear, must be equal to either "N" or "S".

long_shot Must be a valid longitude.

EorW_shot Longitude east or west at the shooting of the gear, must be equal to either "E" or "W".

bot_vshot Must be an integer greater than 0.

date_s The date at the start of a station must be a legitimate date.

Multiple column checks on start date:
The date must fall within the range of the range of the trip start and finish dates.

time_s Start time of the station must be a valid 24-hour time and fall within the range of 0 – 2359 hours.

Multiple column checks on shot date, time and start date, time:
Start date and time must be equal to or not longer than 2 hour after shot date and time.

lat_s Must be a valid latitude

NorS_s Northern or Southern Hemisphere at station start, must be equal to either "N" or "S".

long_s Must be a valid longitude.

EorW_s Longitude east or west at station start, must be equal to either "E" or "W".

gear_s Depth of gear. Must be an integer greater than 0.

bot_gs Depth of sea bottom at the gear's position must be an integer greater than 0, and must not be less than depth of gear

bot_vs Depth of sea bottom at the vessel's position must be an integer greater than 0.

date_f The date at the finish of a station must be a legitimate date.

Multiple column checks on finish date:
The date must fall within the range of the range of the trip start and finish dates.

time_f Finish time of the station must be a valid 24-hour time and fall within the range of 0 – 2359.

Multiple columns checks on date and time:
The start date and time must not be later than the finish date and time, and within a reasonable time period for the type of gear deployed during the station.

lat_f Must be a valid latitude

NorS_f Northern or Southern Hemisphere at station finish, must be equal to either "N" or "S".

long_f Must be a valid longitude.

EorW_f Longitude east or west at station finish, must be equal to either "E" or "W".

Multiple columns checks on position:
The finish position should be within a reasonable distance from the start position for the type of gear deployed during the station.

gear_f	Depth of gear. Must be an integer greater than 0.
bot_gf	Depth of sea bottom at the gear's position must be an integer greater than 0, and must not be less than depth of gear
bot_vf	Depth of sea bottom at the vessel's position must be an integer greater than 0.
min_gdepth	Minimum gear depth must be less than or equal to the depth of gear at the start and finish of the station.
max_gdepth	Maximum gear depth must be greater than or equal to the minimum gear depth and the depth of gear at the start and finish of the station
gear_meth	Gear method code must be a valid code as listed in the *meth_codes* table in the **rdb** database.
gear_code	Must within the range 1 – 4 to relate to gear details in *gear1* to *gear4* respectively in the *t_trip* table.
gear_perf	The gear performance code must be valid code as listed in Appendix 1.
speed	The vessel's recorded speed during the station should be within the range 0 – 5 knots and be reasonable for the gear method.
distance	The distance travelled during the station should be reasonable for the gear method.

Multiple columns check on: distance; start and finish positions; and speed and start/finish times:
The distance travelled during a station as calculated by (1) the difference between start and finish positions; (2) speed * elapsed time; and (3) recorded distance should be in approximate agreement.

wind_dir	Wind direction must fall within the range of 0-359, 999.
wind_force	Wind force must fall within the range of 0 – 12.

air_temp	Air temperature should fall within the reasonable range of 5 – 30.
air_press	Air pressure should fall within the reasonable range of 960 to 1040.
cloud_cov	Cloud cover must fall within the range of 0-8.
sea_cond	The sea condition code must be valid code as listed in Appendix 1.
sea_col	The sea colour code must be valid code as listed in Appendix 1.
swell_ht	The swell height code must be valid code as listed in Appendix 1.
swell_dir	Wind direction must fall within the range of 0-359, 999.
bot_type	The bottom type code must be valid code as listed in Appendix 1.
bot_cont	The bottom contour code must be valid code as listed in Appendix 1.
surf_temp	Sea surface temperature should fall within the reasonable range of 5 – 28.
bot_temp	Sea bottom temperature should fall within the reasonable range of 3 – 25.
wind_spd	Wind speed should fall within the reasonable range of 0 - 30.
secchi	Secchi disc distance should fall within the reasonable range of 0 – 40.
other	Data recorded in the attribute *other* should be accompanied by a matching record in the *t_stat_comm* table.

Station comments (t_stat_comm)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

station_no Must be a unique number within a single trip.

Multiple columns check on trip code and station number:
The combination of trip code and station number must exist in the *t_station* table.

Wire Angles (t_wire_angle)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

station_no Must be a unique number within a single trip.

Multiple columns check on trip code and station number:
The combination of trip code and station number must exist in the *t_station* table.

wire_out Wire out should be within the reasonable range of 0 to 2000

Multiple columns check on trip code, station number, and wire out:
The combination of trip code, station number, and wire out must be unique.

wire_angle Wire angle must be within the range 0 to 90.

Multiple gear units (t_gear_unit)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

station_no Must be a unique number within a single trip.

gear_code Gear code must contain a value

Multiple columns check on trip code, station number, and gear code: The combination of trip code, station number, and gear code must be unique and exist in the *t_station* table.

mesh_size Mesh size should be within the reasonable range of 0.1 to 9.9.

no_bottles The number of bottles should be within the reasonable range 0 to 15.

warp_lgth Warp length should be within the reasonable range of 0 to 2000.

Fish egg abundances (t_eggs)

trip_code Must be equal to a trip code as listed in the *t_trip* table.

station_no Must be a unique number within a single trip.

Multiple columns check on trip code and station number: The combination of trip code and station number must exist in the *t_station* table.

species Must be a valid species code as listed in the *curr_spp* tables in the **rdb** database.

samp_no Sample number must be an integer greater than 0.

percent_samp Percentage sampled must be an integer between the ranges of 1 - 100

stage Must be a valid egg stage for the species as listed in Appendix 1

no_a Must be an integer greater than 0.

Plankton abundances (t_abund)

trip_code	Must be equal to a trip code as listed in the *t_trip* table.
station_no	Must be a unique number within a single trip.
	Multiple columns check on trip code and station number: The combination of trip code and station number must exist in the *t_station* table.
species	Must be a valid species code as listed in the *curr_spp* tables in the **rdb** database.
samp_no	Sample number must be an integer greater than 0.
no_a	Must be an integer greater than 0.

Appendix 1 – Reference Code Tables

Gear performance code

1.	Excellent
2.	Satisfactory, catch unlikely to be reduced by performance
3.	Unsatisfactory, catch probably reduced by malfunction or damage
4.	Unsatisfactory, catch reduced by malfunction or damage

Sea condition code

1.	Calm, glassy	0m
2.	Calm	0 – 0.1m
3.	Smooth	0.1 – 0.5m
4.	Slight	0.5 – 1m
5.	Moderate	1 – 2.5m
6.	Rough	2.5 – 4m
7.	Very rough	4 – 6m
8.	High	6 - 10m
9.	Very high	10 – 15m
10.	Huge	over 15m

Sea colour code

01	Deep blue
02	Blue
03	Light blue
04	Greeny blue
05	Bluey green
06	Deep green
07	Green
08	Yellow green

Swell height code

1	Low	0 – 2m
2	Moderate	2 – 4m
3	Heavy	over 4m

Bottom contour code

1.	Unknown
2.	Smooth/flat
3.	Undulating
4.	Hillocky
5.	Rugged
6.	Very rugged

Bottom type code

1.	Unknown
2.	Mud or ooze
3.	Mud with some sand
4.	Sand
5.	Sand/gravel and shells
6.	Shells (broken)
7.	Gravel
8.	Rock
9.	Coral
10.	Stone
11.	Live shell beds
12.	Mud with broken shells
13.	Sponge beds

Position fix method code

01	Radar
02	Dead reckoning
03	Astrofix
04	Transect marks
05	Radio (RDF)
06	Radar and RDF
07	Satnav
08	Global positioning satellite (GPS)
09	Local knowledge
10	GPX