

Database documentation: tag

K. A. Mackay & B. A. Wood

NIWA Fisheries Data Management
Database Document Series

Revised on 8 November 2004

Contents

<i>Database documentation series</i>	4
2 <i>Tagging Programmes</i>	4
2.1 Sources of tagging data	4
2.2 Data loading and validation.....	5
3 <i>Data structures</i>	6
3.1 Table relationships	6
3.2 Database design.....	9
3.3 Handling orphan tag return records	11
3.4 Multiple releases and returns	11
3.5 Tagging programme requirements and user-defined fields.....	11
4 <i>Table summaries</i>	12
5 <i>tag tables</i>	13
5.1 Table 1: t_project	13
5.2 Table 2: t_releases.....	15
5.3 Table 3: t_returns	19
5.4 Table 4: t_catch	22
5.5 Table 5: t_lfreq.....	23
5.6 Table 6: t_tag_types	24
6 <i>tag Business Rules</i>	25
6.1 Introduction to business rules	25
6.2 Summary of rules	26
7 <i>Acknowledgements</i>	32
8 <i>References</i>	32
<i>Appendix 1 – Reference code tables</i>	33

List of Figures

Figure 1 : Entity Relationship Diagram (ERD) of the tag database.	6
Figure 2 : Expanded ERD of the <i>t_releases</i> and <i>t_returns</i> tables.....	8

Revision History

Version	Change	Date	Responsible
1.0	First release	1993	Brent Wood
1.1	Revision	5 March 2002	Kevin Mackay
1.2	Added Revision History table, changed comment on t_releases.weight to kg from g, D Fisher's instruction.	28 August 2003	Fred Wei
1.3	Updated the document to reflect the length of some previously increased character data types, and existing but unlisted additional attributes.	3 March 2004	David Fisher
1.4	Altered proj_code to char(16), dist to decimal(8,3).	5 August 2004	Fred Wei
1.5	Altered t_lfreq.station_code to char(12).	8 November 2004	Fred Wei

Database documentation series

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

This document provides an introduction to the tag survey database **tag**, and is a part of the database documentation series produced by NIWA. It supersedes the previous documentation by Wood (1993) on this database.

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 **tag** database.

Access to this database is restricted to nominated personnel as specified in the Schedule 6 of the current Data Management contract between the Ministry of Fisheries and NIWA. Any requests for data should in the first instance be directed to the Ministry of Fisheries.

2 Tagging Programmes

2.1 Sources of tagging data

Tagging programmes have been used to provide information on fish and fisheries in New Zealand for many years. Many of these programmes are described in a variety of publications, with summaries of tagging programmes carried out in New Zealand included in Crossland (1982) and Murray (1990). A wide variety of species have been the subject of such studies, including finfish, squid, shellfish and rock lobsters.

To facilitate the storage and access of data collected by a wide range of tagging programmes requires a flexible database structure. This is likely to be more complex than that required for any single programme. A brief overview of some different programmes follows to help illustrate this:

1. The 1980/1981 kahawai programme involved a single target species. The animals were marked and released throughout New Zealand during the two year period. Many were measured when tagged, and samples of animals were also taken to provide additional age/growth information. Only one tag was applied to each animal. Animals were not injected with any biochemical markers. This is an

example of a very straightforward tagging programme, intended to provide information on movement, age/growth and relative levels of effort by method and fisher type (recreational/commercial).

2. The co-operative gamefish tagging program is an ongoing tagging programme run by MAF Fisheries in co-operation with the International Game Fishing Association. This programme has several target species, and it is generally not possible to get an accurate length or weight of the animals tagged. This programme is particularly unusual in that there is no single target species.
3. The bluenose tagging programme run off the Wairarapa coast in 1987 does have a single target species. To tag bluenose without damaging them it is necessary to apply the tag to the animals at the depths they are normally found, generally over 200m deep. To do this, baited hook tag tags attached to lines set in appropriate areas were used. Of the baits/tags which were taken from the lines, it is not possible to say what species (or possibly the sea bed) "took" the tags. The actual species tagged cannot be known, except for those tags which were recaptured.
4. A 1987 snapper tagging programme in Tasman Bay had every tenth fish double tagged to help determine the level of tag shedding which occurred. The fish were also injected with tetracycline to assist with age determination upon recapture.
5. Another snapper tagging programme, this time in the Hauraki Gulf during 1994, had snapper tagged with coded wire tags. All landed snapper within a factory were passed through a electronic scanner to detected the tagged fish. Tag numbers were only known at the time of detection, but not release. A known number of snapper were seeded with tags at the factory to calculate the hit rate of the electronic scanner.

2.2 Data loading and validation

As the data from different tagging programmes has been stored in a variety of formats prior to the establishment of the **tag** database, no standard system has been developed for loading data into the database. Many of the validation rules also vary between tagging programmes so these also are not implemented on the database as a whole. Prior to data being loading to the database, for each tagging programme, appropriate validation rules should be developed and the data checked against these rules.

The referential and range checks listed in the table structures and shown in the Entity Relationship Diagram are the only validation checks imposed.

3 Data structures

3.1 Table relationships

This database contains several tables. The ERD for **tag** (Figure 1) shows the physical data model structure¹ 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 key². This schema is valid regardless of the database system chosen, and it can remain correct even if the Database Management System (DBMS) is changed.

The primary keys on some of the tables in this database are not implemented as required for fully normalised relational database design, but approximate this in a simpler fashion, primarily to facilitate ad hoc queries by users. Individual tagging programmes can appear to have tables of their own if required, by implementing the appropriate views on the main tables.

As the primary key for the *t_releases* and *t_returns* tables are different for different tagging programmes, a primary key is not implemented as such on either table.

Most of the tables in the **tag** database have some attributes, called foreign keys³, which contain standard NIWA fisheries codes, such as *species* and *stage_meth*. These attributes provide links to tables in the **rdb** (research database) database, which contains the definitive list of standard codes. Therefore, an expanded ERD for these tables will follow (Figure 2).

Section 5 shows a listing of all the **tag** tables as implemented by the Empress DBMS. As can be seen in the listing of the tables, a table's primary key has a 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.

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.

¹ Also known as a database *schema*.

² A primary key is an attribute or a combination of attributes that contains an unique value to identify that record.

³ 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.

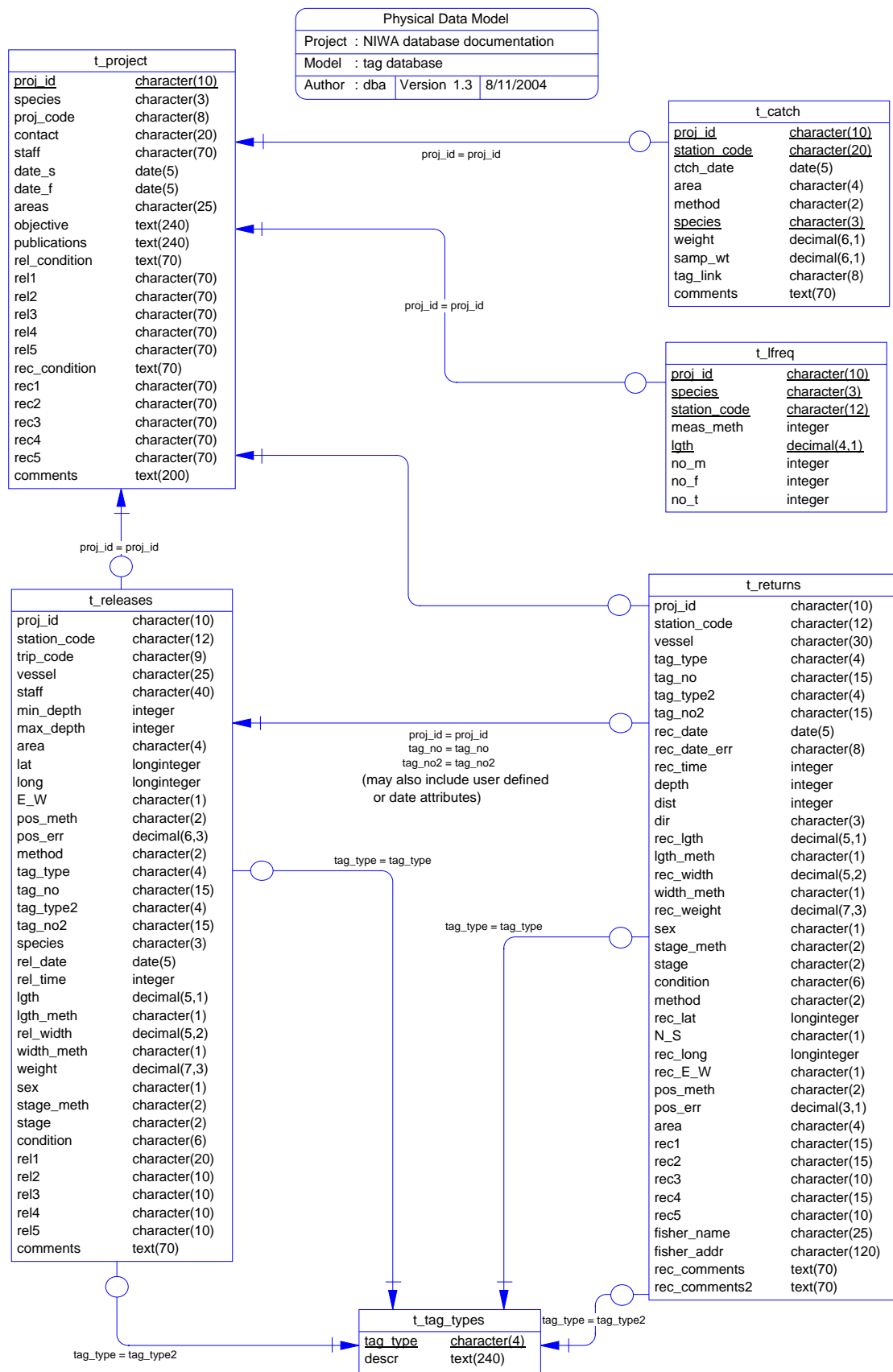


Figure 1 : Entity Relationship Diagram (ERD) of the tag database.

The **tag** 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 **tag** are of the type *one-to-many*⁴. This is shown in the ERD by connecting a single line (indicating ‘many’) from the child table (e.g., *t_catch*) to the parent table (e.g., *t_project*) with an arrow-head (indicating ‘one’) pointing to the parent.

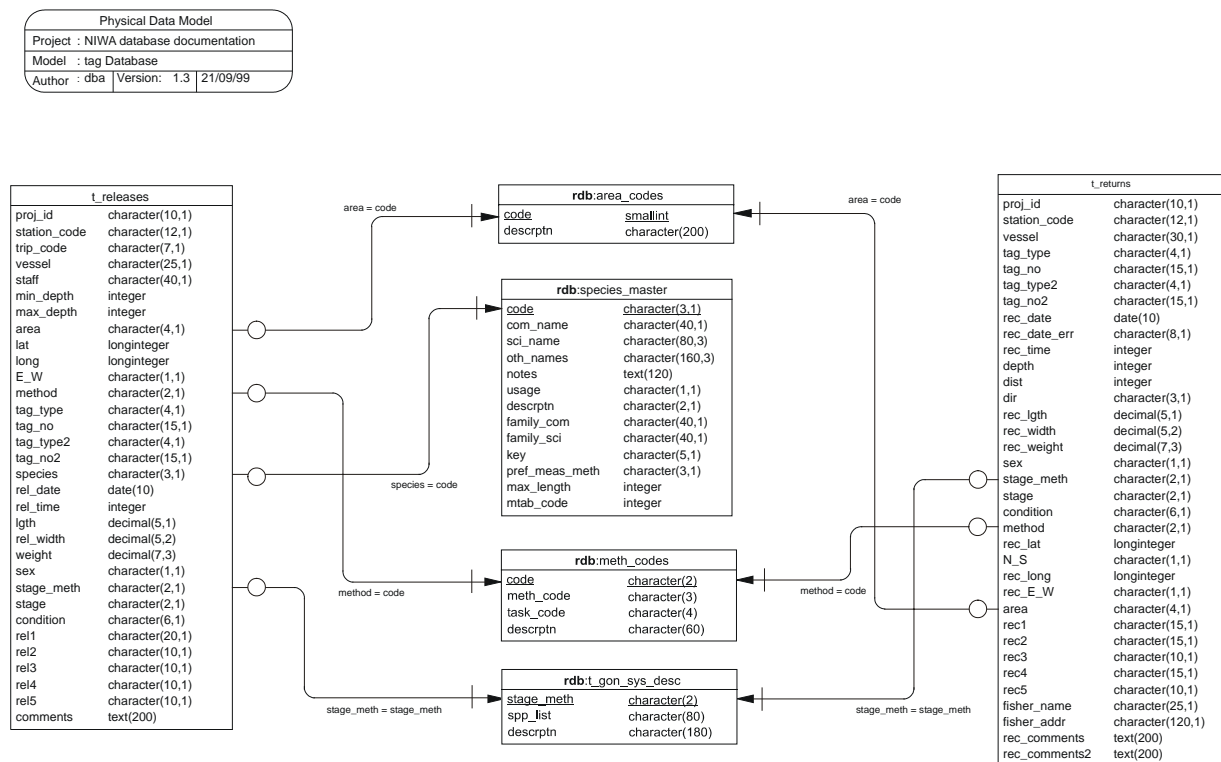


Figure 2 : Expanded ERD of the *t_releases* and *t_returns* tables.

Every relationship has a mandatory or optional aspect to it. That is, if a relationship is mandatory, then it has to occur at least once, while an optional relationship might not occur at all. For example, in Figure 1, consider that relationship between the table *t_project* and it's child table *t_catch*. The symbol “O” by the child *t_catch* means that a tag project can have zero or many catch records, while the bar by the parent *t_catch* means that for every catch record there must be a matching tag project record.

⁴ 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*).

Most of these tables contain foreign keys, which link these tables to each other and to tables in the **rdb** database (Figure 2). The majority of these links are enforced by referential constraints⁵. 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|DELETE
                    parent table (attribute[, attribute])
```

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

```
Referential:      (tag_type) INSERT t_tag_types (tag_type)
```

This means that the value of the attribute *tag_type* in a *t_releases* record must already exist in the parent table *t_tag_type* or the record will be rejected and an error message will be displayed.

A delete constraint implies that for a record to be deleted from a table, the values of the constrained attributes must not be equal to the values of the corresponding attributes in any record of the constraining table. This is used to prevent a parent record from being deleted while child records still exist.

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

The top table shown in the ERD is the *t_project* table (Table 1). Each record stores information describing a tagging programme undertaken on behalf of the MFish. This information does not necessarily relate to data held in other tables in the database, which enables this table to be used as a reference to tagging programmes which have been carried out, even if the actual tagging data is not stored in the database. This table is uniquely keyed on the *proj_id* attribute.

⁵ Also known as integrity checks.

The second table in the ERD, *t_releases* (Table 2), relates to actual animals tagged. Each record represents one animal being marked and released. It is linked to the project table by the *proj_id* attribute, and has a primary key of species, *tag_type* and *tag_no*. To enable programme specific data to be stored in a generic database, five user defined attributes have been implemented. These can each hold up to ten characters and are used if required. The data stored in these attributes is described in the appropriate attributes in the project table.

Under normal circumstances, the *species* attribute in *t_releases* would be linked to the definitive list of species codes in the table *curr_spp*. The *curr_spp* table itself is a view containing only valid species code from the master table *species_master*. However, because species identification of tagged fish generally come from the public and commercial fishers, some obsolete species codes are utilised; e.g. a tagged fish being either a hapuka (HAP) or a bass (BAS), therefore being coded as the obsolete code HPB. Hence the link for species codes is to the master table *species_master*.

Information pertaining to returned tags is kept in the *t_returns* table (Table 3). This is linked to the project table directly via the *proj_id* attribute, and to the releases table via the *tag_type* and *tag_no*. (For programmes where animals were double tagged, a link can also be made using the *tag_type2* and *tag_no2* attributes). This table also has five user defined attributes similar to those in the releases table, but for programme specific recapture or return data.

Some tagging programmes recorded the total catch weight of catches from which animals were tagged or recaptured. These data are kept in the *t_catch* table (Table 4), which is linked to the *t_project* table by the *proj_id* attribute, and may be linked to either the *t_releases* or *t_returns* tables by the *proj_id* and *station_code* attributes.

In conjunction with the actual tagging of animals, some programmes collected length frequency data on catches from which animals were tagged. This is stored in the *t_lfreq* table (Table 5) and may be independent from data in the *t_catch* table. Length frequency data is linked to the *t_project* table by the *proj_id* attribute, and may be linked to either the *t_releases*, *t_returns*, or *t_catch* tables by the *proj_id* and *station_code* attributes.

These last two tables are used for tagging programmes that are not carried out in conjunction with market or catch sampling programmes. In which cases, such catch and length frequency data would be held in such databases as **market**, **trawl**, **scallop**, **rlcs**, etc.

A description of each type of tag used is kept in the *t_tag_types* (Table 6) table, along with a code representing the tag type. The tag type is linked to the *t_releases* and *t_returns* tables by the *tag_type* (and *tag_type2*) attributes.

3.3 Handling orphan tag return records

In strictly logical terms, a tag return record can not exist with out a matching tag release. This would normally be enforced by a referential constraint on *t_returns* using the attributes *proj_id*, *tag_no* and *tag_no2*. However, the reality is that tags get damaged resulting in tag numbers becoming partially or wholly illegible and impossible to be matched against any one release record. In such instances, dummy release records are inserted into the *t_releases* table to match the damaged returned tags.

3.4 Multiple releases and returns

With certain species, such as crayfish, individual tagged animals may be released and recaptured many times. In such cases, one animal is represented by multiple records in the *t_releases* and *t_returns* tables, each with the same keys *proj_id*, *tag_no*, and *tag_no2*, but with different release and return dates.

Care, therefore, must be taken when joining *t_releases* and *t_returns* using the *proj_id*, *tag_no*, and *tag_no2* key, as these cases result in a many-to-many relationship. The tag database schema appears to fail in these instances as many-to-many instances can not be resolved by relational theory.

However, by using one of the user-defined fields for a sequential release and return number, one can resolve this issue. Each time an animal is released, the release number is incremented by one (first release = 1). Similarly, each time the animal is recaptured, the return number is also incremented by one. The keys needed to match a tag return with its appropriate release therefore are: *proj_id*, *tag_no*, *tag_no2*, and release/return number.

Consult the *t_project* table for definitions of the user-defined tables.

3.5 Tagging programme requirements and user-defined fields

By in large, the attributes of the main tag tables (*t_releases* and *t_returns*) are for the main data items that are common for the majority of tagging programmes. However, each programme has certain data fields that are relevant for that specific programme only. Rather than constantly add attributes to these table when the need arises, this database was created with the flexibility of user-defined fields (*rel1 - rel5* and *rec1 - rec5* in the *t_releases* and *t_returns* tables respectively) that will hold any kind of data. Interpretation of these fields will differ from programme to programme, their usage's are defined in the *t_project* table.

4 Table summaries

The **tag** database has six tables containing tag data. The following is a listing and brief outline of the tables contained **tag**:

1. **t_project** : contains details and descriptions of individual tagging projects, including definitions of user-defined fields used in the *t_releases* and *t_returns* tables.
2. **t_releases** : contains details of tagged animal releases.
3. **t_returns** : contains details of tagged animal returns or recaptures.
4. **t_catch** : contains catch data for some tagging projects.
5. **t_lfreq** : contains length frequency data for some tagging projects.
6. **t_tag_types** : contains descriptions of the different types of tags.

5 tag tables

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

5.1 Table 1: t_project

Comment: Table to hold information relating to individual tagging programmes.

Attributes	Data Type	Null?	Comment
proj_id	character(10,1)	No	Unique identifier to distinguish different tagging programmes.
species	character(3,1)	No	The species code for mono-specific tagging programmes, otherwise "MIX".
proj_code	character(16,1)		Project code for the tagging programme (if available).
contact	character(20,1)	No	The staff member who is the main contact regarding the data.
staff	character(70,1)		Staff members involved with the tagging programme.
date_s	date(5)		Start date of the tagging programme
date_f	date(5)		End date of the tagging programme
areas	character(25,1)	No	Four char area codes as found in rdb:area_codes.
objective	text(240,240,480,1)		States the goals/objectives of the programme.
publications	text(240,240,720,1)		Lists publications used to plan or describing data from the programme.
rel_lgth	text(70,70,200,1)		Description of the usage of the t_release lgth field.
rel_width	text(70,70,200,1)		Description of the usage of the t_release rel_width field.
rel_condition	text(70,70,200,1)		Description of the usage of the t_release condition field.
rel1	character(70,1)	No	Description of the usage of the t_release rel1 field. ("not used" if it isn't.)
rel2	character(70,1)	No	As for rel1...
rel3	character(70,1)	No	As for rel1...

Attributes	Data Type	Null?	Comment
rel4	character(70,1)	No	As for rel1...
rel5	character(70,1)	No	As for rel1...
rec_condition	text(70,70,200,1)		Description of the usage of the t_returns condition field.
rec1	character(70,1)	No	Description of the usage of the t_returns.rec1 attribute
rec2	character(70,1)	No	As for rec1...
rec3	character(70,1)		As for rec1...
rec4	character(70,1)		As for rec1...
rec5	character(70,1)		As for rec1...
comments	text(70,70,200,1)		Any text comments to be made on the tagging programme

Creator: dba

Referential: (species) INSERT rdb:species_master (code)

Indices: UNIQUE BTREE project_pk ON (proj_id)

5.2 Table 2: t_releases

Comment: Table to hold information on individual animals/tags released.

Attributes	Data Type	Null?	Comment
proj_id	character(10,1)	No	Identifier to link to project table.
station_code	character(12,1)		Station (or "release site") identifier.
trip_code	character(9,1)		Optional trip code identifier.
vessel	character(25,1)		Name of vessel used to capture the animals for tagging
staff	character(40,1)		Staff involved in this release.
min_depth	integer		Generally either bottom or gear depth as appropriate
max_depth	integer		Generally either bottom or gear depth as appropriate
area	character(4,1)		Area code from rdb:area_codes where release occurred
lat	longinteger		Latitude (as an integer) where release occurred in DDMMmmmm format
long	longinteger		Longitude (as an integer) where release occurred in DDDMMmmmm format
E_W	character(1,1)		East or West longitude
pos_meth	character(2,1)		2 character code for the method of fixing the position. Refer rdb:t_fix_meth_codes
pos_err	decimal(6,3)		Radius of margin of error of the position (nautical miles)
method	character(2,1)		Method used to capture the animals to be tagged.
tag_type	character(4,1)		Code to describe the first (or only) tag used.
tag_no	character(15,1)		Number of the tag used to mark this animal (or number of first tag if two are used.
tag_type2	character(4,1)		Code to describe the second

tag (if used).

Attributes	Data Type	Null?	Comment
tag_no2	character(15,1)		Number on second tag (if used).
species	character(3,1)		3 char species code. Refer rdb:species_master
rel_date	date(5)		Date when release occurred
rel_time	integer		Time of day (24hr) when release occurred
lgth	decimal(5,1)		Length of animal released
lgth_meth	character(1,1)		1 character fish length measurement type code. Refer rdb:t_fish_meas_codes
rel_width	decimal(5,2)		Width of animal released
width_meth	character(1,1)		1 character fish width measurement type code. Refer rdb:t_fish_meas_codes
weight	decimal(7,3)		Weight of animal tagged (may be an estimate) (kg)
sex	character(1,1) range 1 - 4		Sex code: null = not known, 1 = male, 2 = female, 3 = indeterminate or immature, 4 = did not sex
stage_meth	character(2,1)		2 character code to describe gonad staging method. Refer rdb:t_gon_sys_desc
stage	character(2,1)		Stage of sexual maturity (codes vary by species)
condition	character(6,1)		Physical condition of released animal
rel1	character(20,1)		User defined field, as defined in t_project table.
rel2	character(10,1)		User defined field, as defined in t_project table.
rel3	character(10,1)		User defined field, as defined in t_project table.
rel4	character(10,1)		User defined field, as defined in t_project table.
rel5	character(10,1)		User defined field, as defined in t_project table.
comments	text(70,70,200,1)		Any text comments on the released animal

Creator: dba

Referential:

```
(area) INSERT rdb:area_codes (code)
(species) INSERT rdb:species_master (code)
(tag_type2) INSERT t_tag_types (tag_type)
(tag_type) INSERT t_tag_types (tag_type)
(proj_id) INSERT t_project (proj_id)
invalid gonad stage meth code (stage_meth) INSERT
    rdb : t_gon_sys_desc (stage_meth)
invalid measurement code (lgth_meth) INSERT
    rdb : t_fish_meas_codes(fish_meas_code)
invalid measurement width (width_meth) INSERT
    rdb : t_fish_meas_codes fish_meas_code)
```

Indices:

```
NORMAL (2, 15) BTREE releases_species_ndx ON (species)
NORMAL (2, 15) BTREE releases_tag_no_ndx ON (tag_no)
NORMAL (2, 15) BTREE releases_tag_no2_ndx ON (tag_no2)
NORMAL (2, 15) BTREE releases_tag_type_ndx ON (tag_type)
NORMAL (2, 15) BTREE releases_proj_id_ndx ON (proj_id)
```

5.3 Table 3: t_returns

Comment: Table to hold information describing recaptured animals or returned tags.

Attributes	Data Type	Null?	Comment
proj_id	character(10,1)	No	Code to link to t_project and t_release tables.
station_code	character(12,1)		Station code as used in t_releases table.
vessel	character(30,1)		Name of vessel where the recapture occurred.
tag_type	character(4,1)	No	Tag type code as defined in t_tag_types table.
tag_no	character(15,1)		Number of first (or only) tag used.
tag_type2	character(4,1)		Tag type code for second tag (if used).
tag_no2	character(15,1)		Number of second tag (if used).
rec_date	date(5)		Date the animal was recaptured.
rec_date_err	character(8,1)		Describes the "error" in the assigned rec_date.
rec_time	integer		Time of day (24hr) recapture took place (if known)
depth	integer		Depth of water OR gear where recapture occurred
dist	decimal(8,3)		Estimated distance traversed between release and recapture positions
dir	character(3,1)		The direction the recaptured animal traveled.
rec_lgth	decimal(5,1)		Length of recaptured animal
lgth_meth	character(1,1)		1 character fish length measurement type code. Refer rdb:t_fish_meas_codes
rec_width	decimal(5,2)		Width of recaptured animal
width_meth	character(1,1)		1 character fish width measurement type code. Refer rdb:t_fish_meas_codes
rec_weight	decimal(7,3)		Weight of recaptured animal (kg)
sex	character(1,1)		Sex code: null = not known, 1=male, 2=female, 3=indeterminate or immature, 4 = did not sex
	range 1 - 4		

Attributes	Data Type	Null?	Comment
stage_meth	character(2,1)		2 character code to describe gonad staging method. Refer rdb:t_gon_sys_desc
stage	character(2,1)		Stage of sexual maturity (codes vary by species)
condition	character(6,1)		Condition of returned animal.
method	character(2,1)		Code for method used to recapture the animal.
rec_lat	longinteger		Latitude where animal was recaptured, DDMMmmmm format
N_S	character(1,1)		Recapture latitude: North or South of Equator
rec_long	longinteger		Longitude where animal was recaptured, DDDMMmmmm format
rec_E_W	character(1,1)		Recapture longitude: East or West
pos_meth	character(2,1)		2 character code for the method of fixing the position. Refer rdb:t_fix_meth_codes
pos_err	decimal(6,3)		Radius of margin of error of the position (nautical miles)
area	character(4,1)		Area code from rdb:area_codes where recapture occurred
rec1	character(15,1)		User defined field as defined in the t_project table.
rec2	character(15,1)		User defined field as defined in the t_project table.
rec3	character(10,1)		User defined field as defined in the t_project table.
rec4	character(15,1)		User defined field as defined in the t_project table.
rec5	character(10,1)		User defined field as defined in the t_project table.
fisher_name	character(25,1)		Name of person who caught/returned the tag
fisher_addr	character(120,1)		Address of person who caught/returned the tag
rec_comments	text(70,70,200,1)		Text comment(s) on this return
rec_comments2	text(70,70,200,1)		Text comment(s) on this return

Creator: dba
Referential: (tag_type) INSERT t_tag_types (tag_type)
(tag_type2) INSERT t_tag_types (tag_type)
invalid gonad stage meth code (stage_meth) INSERT
rdb : t_gon_sys_desc (stage_meth)
Invalid tag number (proj_id, tag_no, tag_no2)
INSERT t_releases (proj_id, tag_no, tag_no2)
Indices: NORMAL (2, 15) BTREE ON (tag_no)
NORMAL (2, 15) BTREE ON (tag_no2)
NORMAL (2, 15) BTREE ON (proj_id, station_code)

5.4 Table 4: t_catch

Comment: Table to store catch data from tagging stations.

Attributes	Data Type	Null?	Comment
proj_id	character(10,1)	No	Identifier to link to project table.
station_code	character(20,1)	No	Identifier to link to releases table.
species	character(3,1)	No	Valid 3 letter species code. Refer rdb:species_master
weight	decimal(5,1)	No	Weight in kg
samp_wt	decimal(6,1)		Weight of sample used for tagging release/return
tag_link	character(8,1)	No	Flag to mark is catch relates to tag releases of returns smatch "releases returns"
comments	text(70,70,200,1)		Catch comments

Creator: dba

Referential: Invalid project id(proj_id) INSERT project (proj_id)
(species) INSERT rdb : species_master(code)
Invalid area code (area) INSERT rdb:area_codes (code)

Indices: UNIQUE BTREE ON (proj_id, station_code, species)
NORMAL (2, 15) BTREE ON (species)

5.5 Table 5: t_lfreq

Comment: Table to store length frequency data collected during a tagging programme.

Attributes	Data Type	Null?	Comment
proj_id	character(10,1)	No	Identifier to link to project table.
species	character(3,1)	No	Valid 3 letter species code. Refer rdb:species_master
station_code	character(12,1)	No	Identifier to link to releases table.
meas_meth	integer		Code describing the method used to derive the length of the animal.
lgth	decimal(4,1)	No	Length in cm (to 1 mm as decimal if required)
no_m	integer	No	Number of males at this length
no_f	integer	No	Number of females at this length
no_t	integer	No	Total of males, females and unsexed animals at this length

Creator: dba

Referential: Invalid project id (proj_id) INSERT project (proj_id) (species) INSERT rdb : species_master (code)

Indices: NORMAL (2, 15) BTREE ON (species)
UNIQUE BTREE ON (proj_id, species, station_code, lgth)

5.6 Table 6: t_tag_types

Comment: Table to identify all different types of tags used.

Attributes	Data Type	Null?	Comment
tag_type	character(4,1)	No	Four char code for distinct tag types
descr	text(240,20,20,1)		Text description of tag type. (Length, colour, etc)

Creator: dba

Indices: UNIQUE BTREE tag_type_pk ON (tag_type)

6 tag Business Rules

6.1 Introduction to business rules

The following are a list of business rules pertaining to the **tag** database (see Section 2.2 “Data loading and Validation”). A business rule is a written statement specifying what the information system (i.e., any system that is designed to handle rock lobster life cycle data) must do or how it must be structured.

There are three recognized 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. Formula and validation rules are implemented by referential constraints, range checks, and algorithms both in the database and during data validation.

Because of the generalised nature of the tag database schema, business rules can not be defined for data in the user-defined fields. For such fields, business rules are often specified in the definition fields in the *t_project* table.

Validation rules may be part of the preloading checks on the data as opposed to constraints or checks imposed by the database. These rules sometimes state that a value should be within a certain range. All such rules containing the word ‘should’ are conducted by preloading software. The use of the word ‘should’ in relation to these validation checks means that a warning message is generated when a value falls outside this range and the data are then checked further in relation to this value.

6.2 Summary of rules

Tagging project details (t_project)

proj_id	Project identifier, must be unique within the tag database.
species	Must be a valid species code as listed in the <i>species_master</i> table in the rdb database.
proj_code	Project code must be a valid code within the NIWA and/or MFish project management system.
date_s	The start date of the tagging programme must be a legitimate date.
date_f	The finish date of the tagging programme must be a legitimate date.
	Multiple column checks on date: The start date must not be later than the finish date.
areas	Each of the listed area codes must be a valid code as listed in the <i>area_codes</i> table in the rdb database.
rel1 - rel5} rec1 - rec5}	Must be used to describe what values are being stored in the user-defined fields in the <i>t_releases</i> and <i>t_returns</i> tables respectively. Default value - "not used".

Tag type details (t_tag_types)

tag_type	Tag type code must be unique.
-----------------	-------------------------------

Tag release details (t_releases)

proj_id	Project identifier, must be unique within the tag database.
min_depth} max_depth}	Must be a number greater than 0. Multiple column checks on depth: The minimum depth must be less than or equal to the maximum depth.
area	Area code. Must be a valid area code as listed in the <i>area_code</i> table in the rdb database.
lat	Must be a valid latitude ranging from 90 to -90.
long	Must be a valid longitude ranging from 0 to 180.
E_W	Must be equal to either an “E” or a “W”.
method	Must be a valid gear method code as listed in the <i>meth_codes</i> table in the rdb database.
tag_type } tag_type2}	Must be a valid tag type code as listed in the <i>t_tag_types</i> table.
species	Must be a valid species code as listed in the <i>species_master</i> table in the rdb database.
rel_date	Must be a valid date.
rel_time	Must be a valid 24 hour time ranging 0 to 2359.
lgth	Must be a number greater than 0.
lgth_meth	Must be a valid fish measurement code as listed in the <i>t_fish_meas_codes</i> table in the rdb database.
rel_width	Must be a number greater than 0.
width_meth	Must be a valid fish measurement code as listed in the <i>t_fish_meas_codes</i> table in the rdb database.
weight	Must be a number greater than 0.
sex	Sex code. Must be a valid sex code as listed in the <i>t_sex_codes</i> table in the rdb database.
stage_meth	Must be a valid gonad staging system as listed in the <i>t_gon_sys_desc</i> table in the rdb database.

stage Gonad (or life cycle) stage. Must be a valid code as listed in the *t_gon_stg_meth* table in the **rdb** database.

rel1 - rel5 User defined fields. Descriptions of the fields usage must be listed in the *t_project* table.

Tag return details (t_returns)

proj_id	Project identifier, must be unique within the tag database.
tag_type } tag_type2}	Must be a valid tag type code as listed in the <i>t_tag_types</i> table.
tag_no } tag_no2}	Should match values in the <i>t_releases</i> table (but depends on the type of tagging programme).
rec_date	Must be a valid date on or after the initial release of the tagged animal.
rec_time	Must be a valid 24 hour time ranging 0 to 2359.
depth	Must be a number greater than 0.
dist	Must be a number greater than 0.
dir	Should be a valid compass direction involving the characters “N”, “E”, “S”, and “W”; e.g. “ENE”, or an integer representing a valid direction in degrees from 0 to 359.
rec_lgth	Must be a number greater than 0.
lgth_meth	Must be a valid fish measurement code as listed in the <i>t_fish_meas_codes</i> table in the rdb database.
rec_width	Must be a number greater than 0.
width_meth	Must be a valid fish measurement code as listed in the <i>t_fish_meas_codes</i> table in the rdb database.
rec_weight	Must be a number greater than 0.
sex	Sex code. Must be a valid sex code as listed in the <i>t_sex_codes</i> table in the rdb database.
stage_meth	Must be a valid gonad staging system as listed in the <i>t_gon_sys_desc</i> table in the rdb database.
stage	Gonad (or life cycle) stage. Must be a valid code as listed in the <i>t_gon_stg_meth</i> table in the rdb database.
method	Must be a valid gear method code as listed in the <i>meth_codes</i> table in the rdb database.
rec_lat	Must be a valid latitude ranging from 90 to -90.

N_S	Must be equal to either an “N” or a “S”.
rec_long	Must be a valid longitude ranging from 0 to 180.
rec_E_W	Must be equal to either an “E” or a “W”.
area	Must be a valid area code as listed in the <i>area_codes</i> table in the rdb database.
rec1 - rec5	User defined fields. Descriptions of the fields usage must be listed in the <i>t_project</i> table.

Tagging catch details (t_catch)

proj_id	Project identifier, must be unique within the tag database.
station_code	Must be a station code that has been used in either the <i>t_releases</i> or the <i>t_returns</i> table.
species	Must be a valid species code as listed in the <i>species_master</i> table in the rdb database.
weight	Must be a number greater than 0.
samp_wt	Must be a number greater than 0, and less than or equal to <i>weight</i> .
tag_link	Must be equal to either “releases” or “returns”.

Tagging length frequency details (t_lfreq)

proj_id	Project identifier, must be unique within the tag database.
station_code	Must be a station code that has been used in either the <i>t_releases</i> or the <i>t_returns</i> table.
species	Must be a valid species code as listed in the <i>species_master</i> table in the rdb database.
meas_meth	Must be a valid fish (animal) measurement method code as listed in the <i>meas_meth</i> table in the rdb database.
lgth	Must be a number greater than 0.
no_m} no_f} no_t}	Must be a number greater than or equal to 0. Multiple column check on no_t, no_m, and no_f The number in <i>no_t</i> must be equal to or less than the sum of <i>no_m</i> and <i>no_f</i> .

7 Acknowledgements

The authors would like to thank Dave Banks for his editorial contribution to this document.

8 References

Crossland, J. 1982: Tagging of marine fishes in New Zealand. *Fisheries Research Division Occasional Publication No 33*. 19p

Murray, T. 1990: Fish-marking techniques in New Zealand. *American Fisheries Symposium* 7:737--745

Wood, B. A. 1993: Marine Research database documentation. 10. tag. *MAF Fisheries Greta Point Internal Report No. 216* 13p.

Appendix 1 – Reference code tables

Codes for attributes *lgth_meth* and *width_meth* from *rdb:t_fish_meas_codes*

fish_meas_code	description
1	Fork Length
2	Total Length
3	Standard Length
4	Mantle Length (squid)
5	Pelvic Length (rays)
6	Carapace Width
7	Shell Height
8	Shell Length
B	Carapace Length - Orbit to Carapace notch (scampi)
G	Tip of snout to posterior end of dorsal fin (Ghost sharks)
E	Eye to Fork Length (billfish)
J	Lower Jaw to Fork Length (billfish)
O	Orb Length - length across the eye (billfish)
C	Carapace Length - Base of antennal platform to posterior margin
W	Tail Width - as legally defined for red rock lobsters
L	Tail Length - as legally defined for red rock lobsters

stage_meth codes and associated *stage* codes from *rdb: t_gon_stg_meth*

stage_meth	stage	description
CF	MM	Males
CF	BF	Berried female
CF	IF	Immature female
CF	MF	Mature female, setae greater than 6mm
CF	SC	Scattered - spent female
CF	UF	Unidentified stage female
RL	0	Hermaphrodite or indeterminate
RL	1	Male
RL	2	Immature female
RL	3	Mature female, setae greater then 6mm
RL	4	Berried female, no eyes on berry
RL	5	Berried female, eyes in berry visible
RL	6	Spent female, with infertile/unhatched eggs visible
RL	7	Spent female, no eggs visible
RL	9	Female, maturity not determined

Codes for attribute *pos_meth* from rdb:t_fix_meth_codes

fix_meth_code	description
01	Radar
02	Dead reckoning
03	Astrofix
04	Transect marks
05	Radio (RDF)
06	Radar and RDF
07	SatNav
08	Global Positioning Satelite (GPS)
09	Local knowledge
10	GPX