# RIPE Database User Manual: Getting Started

*Andrei Robachevsky*
*Shane Kerr*
*Vesna Manojlovic*
*Samantha Dickinson*

Document ID: ripe-253
Date: 5 September 2002
See also: ripe-252

---

## Table of Contents



---

## 1.0 Introduction

This document is intended for users who have no previous experience with the RIPE Database. It is a hands-on tutorial that walks the reader through the basic concepts and techniques that are needed to use the RIPE Database using examples and exercises.

This document is not intended to be a complete reference. Full information on the RIPE Database may be found in the RIPE Database Reference Manual at:

http://www.ripe.net/ripe/docs/databaseref-manual.html

# 2.0 Objectives

This document should give the reader a basic understanding of the following concepts:

- What the RIPE Database is
- How to get information from the RIPE Database
- How to maintain information in the RIPE Database

# 2.1 The RIPE Database

The RIPE Database is a public database that contains information about registered IP address space and AS numbers, routing policies, and reverse DNS delegations in the RIPE region. It is used for Internet network management.

## 2.1.1 Database objects

Records in the RIPE Database are called "objects". Each object is a list of "attribute-value" pairs displayed in plain text. An example:

```
person:  John Smith
address: Example LTD
         High street 12
         St.Mery Mead
         Essex, UK
phone:   +44 1737 892 004
e-mail:  john.smith@example.com
nic-hdl: JS9-TEST
mnt-by:  EXAMPLE-MNT
remarks: *******************************
remarks: This object is only an example!
remarks: *******************************
changed: john.smith@example.com 20020827
changed: john.smith@example.com 20020828
source:  TEST
```

This is a **person** object for John Smith. The attributes are "person:", "address:", "phone:", and so on. An attribute name always starts on the first column, and ends with a colon. Everything after the colon is the value.

Objects can store information about different resources. For example:

| Network management resource | Object types |
| --- | --- |
| IP address ranges | inetnum, inet6num |
| Routing policies | aut-num, route, etc. |
| Reverse DNS delegations | domain |
| Contact information | person, role |
| Authentication information | mntner |

### 2.1.2 TEST Database

The RIPE NCC provides a test database where users may learn how to use the database software. The TEST Database uses the same software as the RIPE Database but changes in the TEST Database do not affect the RIPE Database. The data in the TEST database is not a copy of the data in the RIPE Database and is provided purely for learning purposes.

All examples below use the TEST Database. However all procedures described are the same for the RIPE Database. In the last section, we will explain what the differences are when using the RIPE Database. Please do not use the RIPE Database for testing purposes. We would also appreciate if you delete all objects you have created in the TEST Database when you have finished performing the exercises described below.

## 2.2 How to get information from the RIPE Database

### 2.2.1 Web queries

The simplest way to get information from the TEST Database is to use a web interface available at:

http://www.ripe.net/perl/test-whois

### 2.2.2 Making simple queries

To query for a particular object the user specifies its "primary key". A primary key is an attribute that uniquely identifies this type of the object.

| Object type | Primary attribute | Example |
| --- | --- | --- |
| inetnum | inetnum | 193.0.0.0 - 193.0.0.255 |
| inet6num | inet6num | 2001:0610:0240::/42 |
| person | nic-hdl: | JS9-TEST |

You will get a reply that includes the object in section 2.1.1.

# 2.3 How to maintain information in the RIPE Database

The RIPE Database is used for storing information about Internet resources. You will need to create objects in the database to document your usage of these resources.

Objects in the RIPE Database must reflect the current state of the resources they describe. Therefore, it is also important to update objects as the details of resources change, or delete objects if resources are no longer used. If IP addresses are assigned to customers, or new staff members are appointed as contacts, it is important to create new objects to reflect this in the database.

Updates to the database are submitted via e-mail. Objects to be created, modified, or deleted are sent to a special address where they get processed automatically. A reply is e-mailed back to the sender with the results of the operation.

If there are any errors when processing the e-mail submission, the e-mail message mailed back to the sender will include an error report. If the report does not help locate the problem, the sender should forward a copy of the original e-mail and the error report to <ripe-dbm@ripe.net>. A RIPE NCC staff member can then help locate the problem.

The following sections describe the process of creating and maintaining objects in the RIPE Database. By the end of the document, you will have learned how to create and protect an object representing a network assignment.

## 2.3.1 Creating objects

The **inetnum** object contains information about registered IP address space: the range of numbers, status, and responsible contacts.

Before this object can be created in the database, information that is referenced by this object must be created in the database first. This requires the creation of the following objects:

1. A **person** object representing a responsible administrative and technical contact for this network, referenced from the "admin-c:" and "tech-c:" attributes of the **inetnum** object.

2. A **mntner** object containing authentication information about who can modify contents of this object, referenced from the "mnt-by:", "mnt-lower:", and "mnt-routes:" attributes of the **inetnum** object. The **mntner** object protects the **inetnum** object.

3. After that we can create the **inetnum** object.

## 2.3.2 Registering contact information

Contact information, such as a phone number and e-mail address, is stored in the **person** object. To create a new **person** object in the database:

1. Copy the **person** object template. The template lists the possible attributes in an object and some information about each attribute. To get the template, type the following in the query

window:

```
-t person
```

You will get a reply that looks something like this:

```
person:  [mandatory] [single]   [lookup key]
address: [mandatory] [multiple] [ ]
phone:   [mandatory] [multiple] [ ]
fax-no:  [optional]  [multiple] [ ]
e-mail:  [optional]  [multiple] [lookup key]
nic-hdl: [mandatory] [single]   [primary/look-up key]
remarks: [optional]  [multiple] [ ]
notify:  [optional]  [multiple] [inverse key]
mnt-by:  [optional]  [multiple] [inverse key]
changed: [mandatory] [multiple] [ ]
source:  [mandatory] [single]   [ ]
```

2. Copy the text into a text editor (e.g. notepad or vi). Delete everything to the right of the colon and fill in attribute values. You must complete the attributes listed as "mandatory." An attribute that is "multiple" can be used more than once in an object.

   You may choose not to complete the optional attributes. However, if you choose not to include optional attributes, you must delete the optional attribute entirely, rather than leaving the value empty.

   Use "AUTO-1" for the "nic-hdl:" attribute, your e-mail address for the "changed:" attribute, and "TEST" for the "source:" attribute.

```
person:  John Smith
address: Example LTD
         High street 12
         St.Mery Mead
         Essex, UK
phone:   +44 1737 892 004
e-mail:  john.smith@example.com
nic-hdl: AUTO-1
remarks: ******************************
remarks: This object is only an example!
remarks: ******************************
changed: john.smith@example.com
source:  TEST
```

3. Send the completed object template in plain text to <test-dbm@ripe.net>.

4. Wait for the acknowledgement to come back from the TEST Database. This may take several minutes. If your update was successful you will get a reply containing something like the following:

```
Your update was SUCCESSFUL.

The following objects were processed.

New OK: [person] JS9-TEST
```

Note the text after the [person] tag. This is the NIC handle of the person, and the "AUTO-1" text is changed to this value. It is guaranteed to be unique and is the primary key of this **person** object. Any references to this **person** object will use this NIC handle.

You can use the new "nic-hdl:" attribute to query for this object. If you do this, you can also see that the "changed:" attribute has had the date of the creation added.

If there was an error, the acknowledgement will indicate failure of the object creation along with the errors encountered. For example, it may contain the following:

```
Part of your update FAILED.
Objects without errors have been processed.
Update FAILED: Syntax error in object
```

### 2.3.3 Registering authentication information

"Authentication" is when you prove that you are who you claim to be. This information is needed to prevent other users from modifying your data. In the database, the information needed for verifying authentication is stored in the **mntner** object (also called the maintainer object). To create a new **mntner** object in the database, do the following:

1. Copy the **mntner** object template. To get the template, along with a detailed description of the meaning and syntax of each allowed attribute, type the following in the query window:

   ```
   -v mntner
   ```

   You will get a reply that looks something like the following:

   ```
   mntner:       [mandatory] [single]   [primary/look-up key]
   descr:        [mandatory] [multiple] [ ]
   admin-c:      [mandatory] [multiple] [inverse key]
   tech-c:       [optional]  [multiple] [inverse key]
   upd-to:       [mandatory] [multiple] [inverse key]
   mnt-nfy:      [optional]  [multiple] [inverse key]
   auth:         [mandatory] [multiple] [ ]
   remarks:      [optional]  [multiple] [ ]
   notify:       [optional]  [multiple] [inverse key]
   mnt-by:       [mandatory] [multiple] [inverse key]
   referral-by:  [mandatory] [single]   [inverse key]
   changed:      [mandatory] [multiple] [ ]
   source:       [mandatory] [single]   [ ]

   The content of the attributes of the mntner class are defined below:

       mntner

        A unique identifier of the mntner object.
        Made up of letters, digits, the character underscore "_",
        and the character hyphen "-"; the first character of a name
        must be a letter, and the last character of a name must be a
        letter or a digit.
   ```

2. As with the **person** object, delete everything to the right of the colon and fill in

attribute values. You must complete the attributes listed as mandatory and should delete optional attributes that you do not use.

Use "TEST-MNT" for the "referral-by:" attribute.

```
mntner:      EXAMPLE-MNT
descr:       Sample maintainer for example
admin-c:     JS9-TEST
tech-c:      JS9-TEST
upd-to:      john.smith@example.com
mnt-nfy:     john.smith@example.com
auth:        MD5-PW $1$WKwrFYYt$.oop28gKMiamE52SVHjyn0
mnt-by:      EXAMPLE-MNT
referral-by: TEST-MNT
changed:     john.smith@example.com
source:      TEST
```

3. You must choose your own mntner value, which is EXAMPLE-MNT in the example. Follow the rules that you received from the "-v mntner" query when choosing the name.

4. For the "admin-c:" and "tech-c:" you should use the value of the "nic-hdl:" in the **person** object created beforehand. The database will not allow you to create a **mntner** object unless this **person** object already exists.

5. The "auth:" attribute begins with a keyword identifying the authentication method and is followed by the authentication information needed to enforce that method. In the example given, the MD5-PW method is used. For both the MD5-PW and CRYPT-PW methods, a password is used to authenticate database operations. To encrypt your password to MD5-PW, you can use the web tools here: http://www.ripe.net/cgi-bin/cgicrypt.pl.cgi

6. Send the completed object template in plain text to <test-dbm@ripe.net>.

7. Wait for the acknowledgement to come back from the database. If your update was successful you will get a reply containing something like the following:

```
Your update was SUCCESSFUL.

The following objects were processed.

New OK: [mntner] EXAMPLE-MNT
```

If there was an error, the acknowledgement will indicate failure of the object creation along with the errors encountered. For example, it may contain the following:

```
Part of your update FAILED.
Objects without errors have been processed.
Update FAILED: Syntax error in object
```

8. The e-mail address in the "mnt-nfy" attribute of the **mntner** will be sent an e-mail of the details of the new object.

You can now query the server and see your new **mntner** object. Type the following in the query window, substituting your mntner name:

```
EXAMPLE-MNT
```

You will get back your new **mntner** object, as well as the **person** object referenced.

```
mntner:      EXAMPLE-MNT
descr:       Sample maintainer for example
admin-c:     JS9-TEST
tech-c:      JS9-TEST
upd-to:      john.smith@example.com
mnt-nfy:     john.smith@example.com
auth:        MD5-PW $1$WKwrFYYt$.oop28gKMiamE52SVHjyn0
mnt-by:      EXAMPLE-MNT
referral-by: TEST-MNT
changed:     john.smith@example.com 20020827
source:      TEST

person:      John Smith
address:     Example LTD
             High street 12
             St.Mery Mead
             Essex, UK
phone:       +44 1737 892 004
e-mail:      john.smith@example.com
nic-hdl:     JS9-TEST
remarks:     *******************************
remarks:     This object is only an example!
remarks:     *******************************
changed:     john.smith@example.com 20020827
source:      TEST
```

Be default, a query returns the contact information associated with an object. This is why the **person** object is returned. If you do not want the referral contact information, use the "disable recursion" flag on the query, "-r". You can see this by typing the same query into the query window, putting the flag in front:

```
-r EXAMPLE-MNT
```

Now you will get only the **mntner** object. Disabling recursion can result in a smaller, easier to understand reply if you do not care about contact information. This is often the case when managing your own objects.

## 2.3.4 Protecting your contact information

Now that you have a **mntner**, you can protect objects in the database. An object is protected by a mntner if it references the mntner in the "mnt-by:" attribute. Only a mntner listed as "mnt-by:" is authorised to make changes to an object.

Most objects types require that you protect them with your mntner. However, **person** objects do not. It is recommended that you protect them. To protect your **person** object:

1. Get a copy of your current **person** object. In the query window type the "nic-hdl:" of your **person** object:

```
JS9-TEST
```

You will get back your current object in the database.

You can also search by typing in a name. In this case, the database will return all **person** objects that have that name. For common names there may be many objects returned.

2. Copy the object returned by the query.

```
person:  John Smith
address: Example LTD
         High street 12
         St.Mery Mead
         Essex, UK
phone:   +44 1737 892 004
e-mail:  john.smith@example.com
nic-hdl: JS9-TEST
remarks: *******************************
remarks: This object is only an example!
remarks: *******************************
changed: john.smith@example.com 20020827
source:  TEST
```

3. Add your mntner as the "mnt-by:" for your **person** object. The database will not allow you to use a "mnt-by:" unless the **mntner** object already exists.

4. Add a "changed:" line to reflect the fact that you are updating the object.

```
person:  John Smith
         address: Example LTD
         High street 12
         St.Mery Mead
         Essex, UK
phone:   +44 1737 892 004
e-mail:  john.smith@example.com
nic-hdl: JS9-TEST
remarks: *******************************
remarks: This object is only an example!
remarks: *******************************
mnt-by:  EXAMPLE-MNT                      # new "mnt-by:" attribute
changed: john.smith@example.com 20020827
changed: john.smith@example.com           # new "changed:" attribute
source:  TEST
```

5. When you add a mntner to an object that does not have one, you must authenticate yourself as the new mntner. Since the example uses the MD5-PW method, add a password line to your e-mail. This must start on the first column but can occur anywhere within the body of the message:

```
password: your_cleartext_password_here
```

6. Send the updated object template to <test-dbm@ripe.net>.

7. Wait for the acknowledgement to come back from the database. It will indicate

success or failure of your update.

### 2.3.5 Locating network assignments

Network assignments are represented by **inetnum** objects. Before you can create a new **inetnum** you must find a range of IP addresses that are not currently assigned. This section describes how you can query the database for this information. You can also use the queries for any other purpose when you want to get IP address information from the database.

By default, the database returns the smallest range that includes the entire range queried. This is a "less specific" query. For example, if you query the following:

```
10.11.12.0 - 10.11.13.255
```

You might get something like this:

```
inetnum:     10.0.0.0 - 10.255.255.255
netname:     IANA-ABLK-RESERVED1
descr:       Class A address space for private internets
descr:       See http://www.ripe.net/db/rfc1918.html for details
country:     NL
admin-c:     role1-TEST
tech-c:      role1-TEST
status:      ALLOCATED UNSPECIFIED
remarks:     Country is really worldwide
remarks:     This network should never be routed outside an enterprise
remarks:     See RFC1918 for further information
mnt-by:      TEST-DBM-MNT
mnt-lower:   TEST-DBM-NONE-MNT
mnt-routes:  TEST-DBM-NONE-MNT
changed:     ripe-dbm@ripe.net 20020902
source:      TEST
```

This is the less specific match. The 10.11.12.0 - 10.11.13.255 **inetnum** fits entirely within the 10.0.0.0 - 10.255.255.255 **inetnum**. This is the smallest such block.

If you want the server to give you only an exact match, then you can request this using the "-x" flag. An exact match is one where the IP range of the **inetnum** are the same as the IP range of the query.

```
-x 10.11.12.0 - 10.11.13.255
```

In this case you will get only an exact match, or an error indicating that no such **inetnum** exists:

```
%ERROR:101: no entries found
%
% No entries found in the selected source(s).
```

Sometimes you want to see all of the less specific **inetnum** to a range. In this case, you can use the "-L" flag. If you do this, you will see all **inetnum** that include the entire range queried. For example, if you query the following:

```
-L 10.11.12.0 - 10.11.13.255
```

You might get something like this:

```
inetnum:    0.0.0.0 - 255.255.255.255
netname:    IANA-BLK
descr:      The whole IPv4 address space
country:    NL
admin-c:    role1-TEST
tech-c:     role1-TEST
status:     ALLOCATED UNSPECIFIED
remarks:    Country is really worldwide
remarks:    This address space is assigned at various
remarks:    other places in the world.
mnt-by:     TEST-DBM-MNT
mnt-lower:  TEST-DBM-NONE-MNT
mnt-routes: TEST-DBM-NONE-MNT
changed:    ripe-dbm@ripe.net 20010418
source:     TEST

inetnum:    10.0.0.0 - 10.255.255.255
netname:    IANA-ABLK-RESERVED1
descr:      Class A address space for private internets
descr:      See http://www.ripe.net/db/rfc1918.html for details
country:    NL
admin-c:    role1-TEST
tech-c:     role1-TEST
status:     ALLOCATED UNSPECIFIED
remarks:    Country is really worldwide
remarks:    This network should never be routed outside an enterprise
remarks:    See RFC1918 for further information
mnt-by:     TEST-DBM-MNT
mnt-lower:  TEST-DBM-NONE-MNT
mnt-routes: TEST-DBM-NONE-MNT
changed:    ripe-dbm@ripe.net 20020902
source:     TEST
```

You can also look for smaller **inetnum** contained within a given range. This is a "more specific" query. You can use this on an allocation to look for ranges that have no other assignments. To do this, use the "-m" flag:

```
-m 10.0.0.0 - 10.255.255.255
```

You will get a reply that resembles something like this:

```
inetnum:    10.11.13.0 - 10.11.13.255
netname:    Example-Network
descr:      This is a fictitious assignment for the end-user "Example"
country:    GB
admin-c:    JS9-TEST
tech-c:     JS9-TEST
status:     ASSIGNED PA
notify:     john.smith@example.com
mnt-by:     EXAMPLE-MNT
mnt-lower:  EXAMPLE-MNT
mnt-routes: EXAMPLE-MNT
changed:    john.smith@example.com 20020827
source:     TEST

inetnum:    10.11.11.0 - 10.11.11.255
netname:    Example-Network-2
```

```
descr:       This is a fictitious assignment for the end-user "Example"
country:     GB
admin-c:     JS9-TEST
tech-c:      JS9-TEST
status:      ASSIGNED PA
notify:      john.smith@example.com
mnt-by:      EXAMPLE-MNT
mnt-lower:   EXAMPLE-MNT
mnt-routes: EXAMPLE-MNT
changed:     john.smith@example.com 20020903
source:      TEST
```

This is a "one-level more specific" query. This means that the largest **inetnum** that are within the given range are returned.

In this example the IP addresses 10.11.12.0 - 10.11.12.255 are not assigned and are available. You will need to find an available range to be able to do the next exercise.

If you want to see all **inetnum** smaller than a given range, you can use the "-M" flag:

```
-M 10.0.0.0 - 10.255.255.255
```

This will return all levels of **inetnum** in the range. This can return an extremely large number of objects, but can be useful for finding all of the **inetnum** for a portion of the Internet.

### 2.3.6 Recording network assignments

Now that all of the objects necessary for an **inetnum** have been created and protected and you have located an appropriate range of IP numbers, you can create the **inetnum** object itself. To create a new **inetnum** in the database:

1. Copy the **inetnum** object template. To get the template, type the following in the query window:

   ```
   -t inetnum
   ```

   You will get a reply that looks something like the following:

   ```
   inetnum:     [mandatory] [single]   [primary/look-up key]
   netname:     [mandatory] [single]   [lookup key]
   descr:       [mandatory] [multiple] [ ]
   country:     [mandatory] [multiple] [ ]
   admin-c:     [mandatory] [multiple] [inverse key]
   tech-c:      [mandatory] [multiple] [inverse key]
   rev-srv:     [optional]  [multiple] [inverse key]
   status:      [mandatory] [single]   [ ]
   remarks:     [optional]  [multiple] [ ]
   notify:      [optional]  [multiple] [inverse key]
   mnt-by:      [mandatory] [multiple] [inverse key]
   mnt-lower:   [optional]  [multiple] [inverse key]
   mnt-routes: [optional]  [multiple] [inverse key]
   mnt-irt:     [optional]  [multiple] [inverse key]
   changed:     [mandatory] [multiple] [ ]
   source:      [mandatory] [single]   [ ]
   ```

2. Delete everything to the right of the colon and fill in attribute values. You must complete the attributes listed as mandatory and should delete optional attributes that you do not use.

Use "ASSIGNED PA" for the "status:" attribute, and your e-mail address for the "notify:" attribute. The e-mail address specified in the "notify:" attribute will get mailed when the object changes. You must choose your own "netname:", using the same rules as you did to choose a mntner name.

```
inetnum:     10.11.12.0 - 10.11.12.255
netname:     Example-Network
descr:       This is a fictitious assignment for the end-user "Example"
country:     GB
admin-c:     JS9-TEST
tech-c:      JS9-TEST
status:      ASSIGNED PA
notify:      john.smith@example.com
mnt-by:      EXAMPLE-MNT
mnt-lower:   EXAMPLE-MNT
mnt-routes:  EXAMPLE-MNT
changed:     john.smith@example.com
source:      TEST
```

3. When a new object is created that has a "mnt-by:" attribute, the mntner must authorise the creation. Add the appropriate password for the mntner in the "mnt-by:" attribute:

```
password: your_cleartext_password_here
```

4. Send the completed object template in plain text to <test-dbm@ripe.net>.

5. Wait for the acknowledgement to come back from the database. If your update was successful you will get a reply containing something like the following:

```
Your update was SUCCESSFUL.

The following objects were processed.

New OK: [inetnum] 10.11.12.0 - 10.11.12.255
```

If there was an error, the acknowledgement will indicate failure of the object creation along with the errors encountered. For example, it may contain the following:

```
Part of your update FAILED.
Objects without errors have been processed.
Update FAILED: Syntax error in object
```

6. The e-mail address in the "mnt-nfy" attribute of the mntner will be sent an e-mail of the details of the new object.

### 2.3.7 Updating the inetnum object

Suppose you later need to update information in your **inetnum** object. For instance, the technical contact has changed and is now represented by the **person** object "JJ1-TEST". (You must create a new **person** object before you can follow this example.) To update an existing object, do the following:

1. Query the RIPE Database for the object.

```
10.11.12.0 - 10.11.12.255
```

2. Copy the object returned by the query.

```
inetnum:    10.11.12.0 - 10.11.12.255
netname:    Example-Network
descr:      This is a fictitious assignment for the end-user "Example"
country:    GB
admin-c:    JS9-TEST
tech-c:     JS9-TEST
status:     ASSIGNED PA
notify:     john.smith@example.com
mnt-by:     EXAMPLE-MNT
mnt-lower:  EXAMPLE-MNT
mnt-routes: EXAMPLE-MNT
changed:    john.smith@example.com 20020828
source:     TEST
```

3. Change the "tech-c:" attribute. Add a "notify:" attribute so the new technical contact will be notified when the **inetnum** is modified.

```
inetnum:     10.11.12.0 - 10.11.12.255
netname:     Example-Network
descr:       This is a fictitious assignment for the end-user "Example"
country:     GB
admin-c:     JS9-TEST
tech-c:      JJ1-TEST # changed to new nic-hdl
status:      ASSIGNED PA
notify:      john.smith@example.com
notify:      jan.jansen@example.com                 # added new notify
mnt-by:      EXAMPLE-MNT
mnt-lower:   EXAMPLE-MNT
mnt-routes:  EXAMPLE-MNT
changed:     john.smith@example.com 20020828
changed:     john.smith@example.com                 # new changed line
source:      TEST
```

Please note that you cannot change the primary attribute of the object (inetnum: 10.11.12.0 - 10.11.12.255). The database will consider this to be a creation of a new object.

4. To change an object that is protected by a "mnt-by:" attribute, you must add the appropriate authentication:

```
password: your_cleartext_password_here
```

5. Send the updated object template to <test-dbm@ripe.net>

6. Wait for the acknowledgement to come back from the database. It will indicate success or failure of your update.

7. The e-mail address in the "notify:" attribute of the original object will be sent a message with the details of the change.

### 2.3.8 Deleting objects

Sometimes you no longer need objects you maintain. These should be deleted. For example, if the assignment is no longer used you should delete the **inetnum** object and all **person** objects that are only referenced from that object.

To delete an existing object:

1. Query the database for your object.

2. Copy the object returned by the query.

3. Add a special "delete:" attribute to the object explaining why the object should be deleted. For example:

```
inetnum:     10.11.12.0 - 10.11.12.255
netname:     Example-Network
descr:       This is a fictitious assignment for the end-user "Example"
country:     GB
admin-c:     JS9-TEST
tech-c:      JJ1-TEST
status:      ASSIGNED PA
notify:      john.smith@example.com
notify:      jan.jansen@example.com
mnt-by:      EXAMPLE-MNT
mnt-lower:   EXAMPLE-MNT
mnt-routes:  EXAMPLE-MNT
changed:     john.smith@example.com 20020828
changed:     john.smith@example.com 20020829
source:      TEST
delete:      assignment returned by customer
```

4. To delete an object that is protected by a "mnt-by:" attribute, you must add the appropriate password:

```
password: your_cleartext_password_here
```

5. Send the object to be deleted to <auto-dbm@ripe.net>.

6. Wait for the acknowledgement to come back from the database. It will indicate success or failure of your deletion.

7. The e-mail addresses in the "notify:" attribute of the object will be sent a message with the details of the deletion.

Objects that are referenced by other objects cannot be deleted. For example, a **mntner** object cannot be deleted while is used as an "mnt-by:" or a "mnt-lower:". You can find the references to a **mntner** object by using an inverse query. Type the following in the query window, substituting your **mntner** object:

```
-i mnt-by,mnt-lower,mnt-routes -r EXAMPLE-MNT
```

This will return all of the objects that reference EXAMPLE-MNT. The "-i" flag requests the inverse query, and the "mnt-by,mnt-lower,mnt-routes" specify which attributes you want to look for. There must not be a space after the comma. The "-r" disables recursion, as seen in section 2.3.3.

Since an organisation usually uses one mntner, you can use this query to locate all of the objects for an organisation.

Before you can delete a mntner, you must remove all references to it. For example, if you have the following **mntner** and **person**:

```
mntner:      EXAMPLE-MNT
descr:       Sample maintainer for example.
admin-c:     JS9-TEST
tech-c:      JS9-TEST
upd-to:      john.smith@example.com
mnt-nfy:     john.smith@example.com
auth:        MD5-PW $1$WKwrFYYt$.oop28gKMiamE52SVHjyn0
mnt-by:      EXAMPLE-MNT
referral-by: TEST-MNT
changed:     john.smith@example.com 20020827
source:      TEST

person:      John Smith
address:     Example LTD
             High street 12
             St.Mery Mead
             Essex, UK
phone:       +44 1737 892 004
e-mail:      john.smith@example.com
nic-hdl:     JS9-TEST
remarks:     ******************************
remarks:     This object is only an example!
remarks:     ******************************
mnt-by:      EXAMPLE-MNT
changed:     john.smith@example.com 20020827
changed:     john.smith@example.com 20020828
source:      TEST
```

The **mntner** "EXAMPLE-MNT" cannot be deleted because it is referenced by the **person** "JS9-TEST", and the **person** "JS9-TEST" cannot be deleted because it is referenced by the **mntner** "EXAMPLE-MNT". To delete these objects, do the following:

1. Update the **person** object, and remove the "mnt-by:" attribute. This removes all protection, but this is not a security issue because the object will be deleted soon.

2. Delete the **mntner** object.

3. Delete the **person** object.

(Please remember to delete all objects you created in the TEST Database while doing these exercises.)

## 2.4 Using the production RIPE Database

You should now have an understanding of the basic concepts of the RIPE Database and be able to maintain your own data and perform queries. This section details the differences between the TEST Database and the RIPE Database.

1. Queries use a different search tool:

   http://www.ripe.net/perl/whois

2. Objects in the RIPE Database use RIPE instead of TEST for both the "source:" attribute and the suffix appended to "nic-hdl:" attributes.

3. Updates to the RIPE Database should be sent to <auto-dbm@ripe.net> rather than <test-dbm@ripe.net>.

4. You cannot create your own **mntner** object in the RIPE Database. You should send your update to <auto-dbm@ripe.net> but it will be processed by a RIPE NCC staff member. **mntner** objects are only created for users who are referenced as the "admin-c:" or "tech-c:" for **inetnum**, **inet6num**, **aut-num**, or **domain** objects.

5. You should not create **inetnum** objects in the RIPE Database unless you have received authorisation by the LIR that holds the responsibility for that address range.

## 2.5 Where to learn more

The following resources are available to help you as you use the RIPE Database.

### 2.5.1 Whois help

A query for "help" will return a full list of all of the flags that you can use to query the database.

```
help
```

While some of these have been covered in this document, many have not.

### 2.5.2 Database FAQ

The Database Frequently Asked Questions (FAQs) is available at:

   http://www.ripe.net/ripencc/faq/database/index.html

This is updated with helpful information based on the needs of the users and new features offered by the software.

### 2.5.3 RIPE Database Reference Manual

The definitive source of information for the RIPE Database is the RIPE Database Reference Manual. The latest version of this is available from the RIPE Document Store at:

http://www.ripe.net/ripe/docs/databaseref-manual.html

It contains detailed specifications on all of the topics covered in this guide, as well as every aspect of using the RIPE Database.

### 2.5.4 LIR Training Courses

The RIPE NCC provides training for Local Internet Registries. More information about the RIPE NCC LIR Training Courses can be found at:

http://www.ripe.net/training/

### 2.5.5 Specific questions

If you have a specific question that has not been answered in this guide, you may post it to the <db-help@ripe.net> mailing list. Details about this list can be found here:

http://www.ripe.net/ripe/mail-archives/db-help/index.html

You can also e-mail your question to <ripe-dbm@ripe.net>