# Generate Structures Mgr Users for HE

Contents

[Generate Structures Mgr Users for HE 1](#_Toc3907032)

[Overview 1](#_Toc3907033)

[Executing the Process 1](#_Toc3907034)

[Elements of the Process 1](#_Toc3907035)

[bulk\_user\_creation.sql 1](#_Toc3907036)

[Str\_user\_data.sql 2](#_Toc3907037)

[Generate\_hestrusers package 2](#_Toc3907038)

[Migrate\_buc\_to\_sso.sql 2](#_Toc3907039)

## Overview

This process requires a table to be created and populated from the spreadsheet of trained users from Kahootz. It then works through the table checking the suggested username and initials against what already exists, assigns new ones if necessary, and creates the users. Any errors are reported in the table for review after the process has completed.

## Executing the Process

The steps are:

1. Create table bulk\_user\_creation using the script bulk\_user\_creation.sql
2. Populate the table using str\_user\_data.sql
3. Create package by running generate\_hestrusers.pkh and generate\_hestrusers.pkb
4. Execute this script to create the users:

begin

generate\_hestrusers.create\_users;

end;

/

1. Migrate the successful users to SSO by running the migrate\_buc\_to\_sso.sql script.
2. Review the bulk\_user\_creation table for any error records and sort them out manually. Most will be existing users or clashes where user’s email address has changed but the name is the same. If any new users are created don’t forget SSO settings. Assign lb\_user to any existing users.
3. Re-insert any removed apostrophes manually through the users form.
4. Use this query to extract the user data for whoever’s creating the users in eB:

SELECT buc\_name name,

buc\_email\_address email,

buc\_actual\_username username,

buc\_actual\_initials initials,

lower(hus\_initials) || hus\_user\_id password,

buc\_au1 au1,

buc\_au2 au2,

buc\_au3 au3,

buc\_au4 au4

FROM hig\_users, bulk\_user\_creation

WHERE hus\_username = buc\_actual\_username;

## Elements of the Process

### bulk\_user\_creation.sql

The table bulk\_user\_creation is created using bulk\_user\_creation.sql and is used to store the user data from training’s spreadsheet.

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| BUC\_NAME | VARCHAR2(30) | Populated from spreadsheet. User’s name. Filter out any apostrophes. |
| BUC\_INITIALS | VARCHAR2(3) | Suggested initials for user |
| BUC\_EMAIL\_ADDRESS | VARCHAR2(255) | Populated from spreadsheet. |
| BUC\_USERNAME | VARCHAR2(30) | Suggested username for user |
| BUC\_AU1 | VARCHAR2(10) | Populated from spreadsheet, first admin unit listed |
| BUC\_AU2 | VARCHAR2(10) | Populated from spreadsheet, second admin unit listed |
| BUC\_AU3 | VARCHAR2(10) | Populated from spreadsheet, third admin unit listed |
| BUC\_AU4 | VARCHAR2(10) | Populated from spreadsheet, fourth admin unit listed |
| BUC\_TRAIN\_DATE | DATE | Populated from spreadsheet. If blank then defaults to sysdate |
| BUC\_ACTUAL\_USERNAME | VARCHAR2(30) | Created during process, can be the same as suggested if there isn’t a clash otherwise suitable alternative used |
| BUC\_ACTUAL\_INITIALS | VARCHAR2(3) | Created during process, can be the same as suggested if there isn’t a clash otherwise suitable alternative used |
| BUC\_ERROR | VARCHAR2(1000) | Only populated if an error occurs during the process. |

### Str\_user\_data.sql

Insert statements generated from training’s spreadsheet. Populates bulk\_user\_creation with the basic data needed to create the users. I’ve already worked through the current spreadsheet removing apostrophes and writing up suggested usernames and initials to use as a test exercise. Once the spreadsheet is finalised I will complete script str\_user\_data.sql.

### Generate\_hestrusers package

The package containing the code to create the users. Assigns admin units, roles and user options.

If the suggested username from the spreadsheet is already in use, it adds a number to the end (up to 99) until a free username is found.

Similarly if the suggested initials are already in use, it uses the first 2 characters plus a digit, or if they’re all in use too, the first character plus 2 digits until it finds a value not already in use.

Errors should all be trapped and written to the buc\_error column of the table rather than causing the procedure to stop so that it attempts to process all users.

### Migrate\_buc\_to\_sso.sql

Adapted version of the standard SSO migration script to migrate any successfully created users to be SSO/IMS users. It will only pick up those with no error message.