



AAiT

ADDIS ABABA INSTITUTE OF TECHNOLOGY

አዲስ አበባ ቴክኖሎጂ አጠቃላይ

ADDIS ABABA UNIVERSITY

አዲስ አበባ ዘመን

Department of Information Technology & Scientific Computing (Software Engineering)

Course name: - Big Data Analysis
Lab Report - I

Prepared By: -

Abdulhamid Abdo	ATR/8526/07
Abrham Kindu	ATR/6689/07
Samuel Mussie	ATR/9258/07

Submitted to: - Alazar Alemayehu

Dec 2018

ABSTRACT

Connecting to a database from the R environment can be extremely useful. It allows us direct access to the data without have to first export it from a database and then import it from a csv file or entering it directly into R. This assignment documentation will showcase the processes and steps to set up R to connect to a database. This documentation will show step-by-step instructions and provides the R code on how to query the database once connectivity is established.

Introduction

We all know that R is a very powerful when working with big data. We can manipulate and deal with small datasets that easily fit into your computer's memory. But what if we encounter datasets that are too large for your computer to handle as a whole? In this case, storing the data outside of R and organizing it in a database will become handy. Connecting to the database allows you to retrieve only the chunks needed for the current analysis. R can connect to almost any existing database type. Most common database types have R packages that allow you to connect to them. Examples could be RSQLite, RMySQL, RODBC, etc.). We will be using RODBC to connect to our database and perform CRUD operations on our database.

Problem Description

Performing CRUD (Create, Read, Update, Delete) operations using R to a database.

(A) Design a simple database containing 3 tables with relationships.

- We have created a database called 'BIGDATA' with three tables namely 'USER', 'GRADES', and 'ADDRESS'.
- USER table basic information of a user which has 'id', 'first_name', 'last_name' and 'sex' attributes.
- GRADES table contains information about a user's grade and has 'id', 'user_id', 'course_name' and 'course_grade' attributes. It has 'many to many' relationship with table 'USER' as many grades can be possessed by a user.
- ADDRESS table contains information about a user's address and has 'id', 'user_id', 'house_no', 'region' and 'city' attributes. It has 'one to one' relationship with table 'USER' assuming a user has only one address to live on.

The ER diagram for the database is shown below.

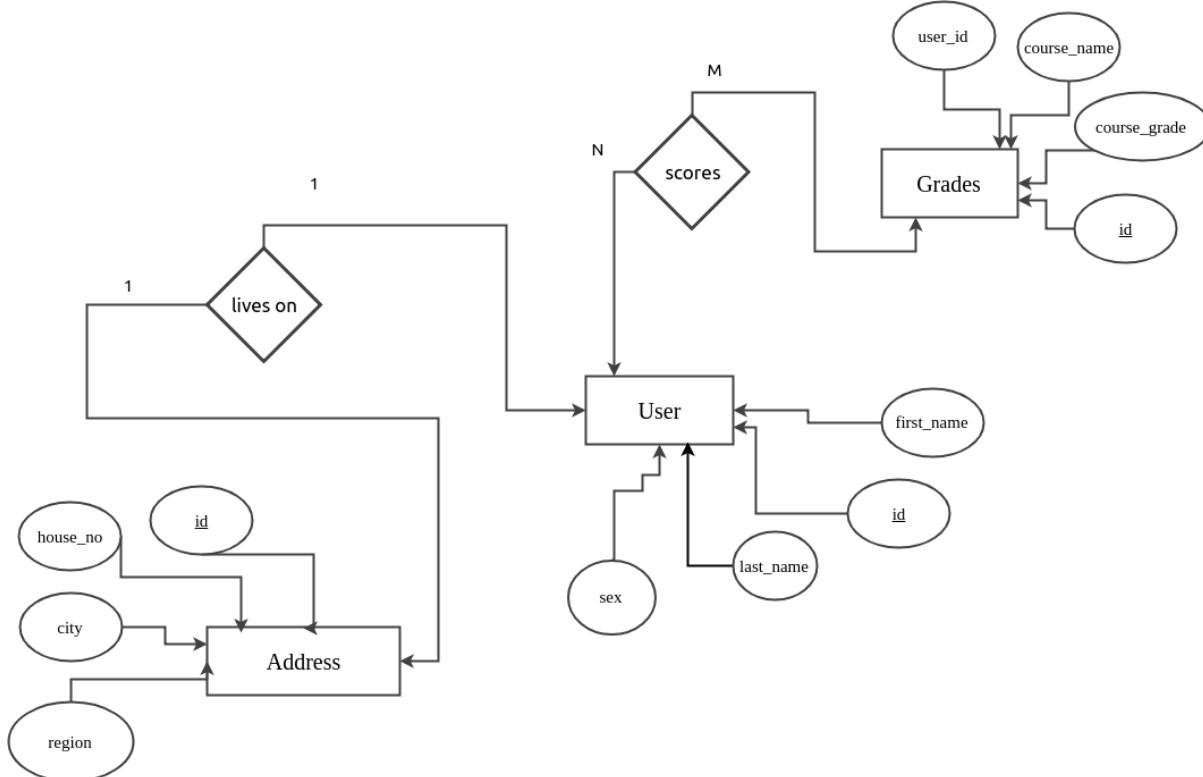
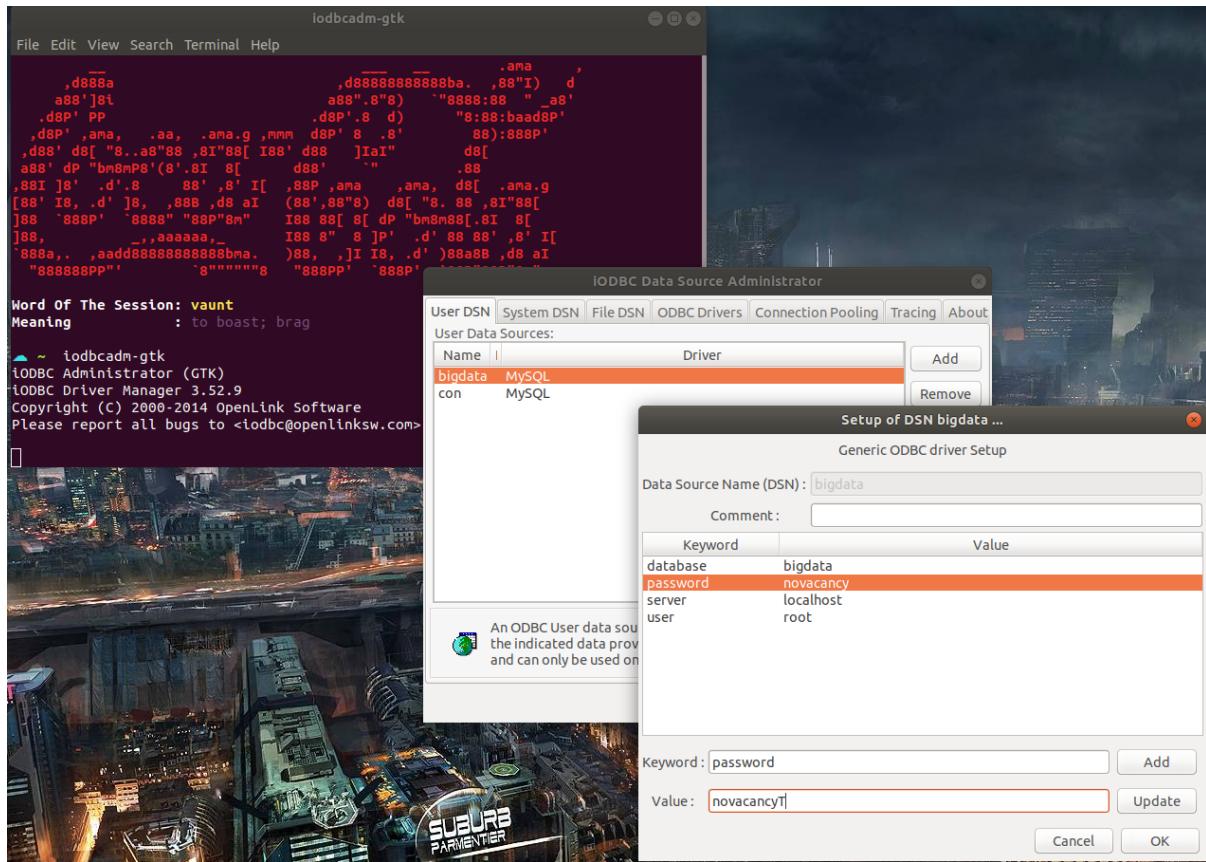


Figure 1 ER diagram for database BIGDATA

- Setting up a DSN named 'bigdata' and configuring a database also named bigdata.



(B, C, D & E) Create a table from R program and demonstrate CRUD operations.

- We will be showing step by step processes of creating the tables by using screenshots.

The screenshot shows an RStudio session with an R script file named 'CRUDDB.R'. The code performs the following steps:

- Sets the working directory to 'Desktop/Class/IODATA/Assignment/'.
- Loads the 'RODBC' library.
- Creates a connection to the 'bigdata' database using 'odbcConnect()' with 'uid = "root"' and 'pwd = "novacancy"'.
- Checks if the 'users' table exists in the connection.
- If it doesn't exist, creates it with columns 'id' (int), 'first_name' (varchar(25)), and 'last_name' (varchar(25)).
- Adds primary key 'id' to the 'users' table.
- Inserts three rows of data into the 'users' table: ('Samuel', 'Mussie'), ('Abrahem', 'Kindu'), and ('Abdulhamid', 'Abdo').
- Selects all data from the 'users' table.
- Shows the contents of the 'users' table in the RStudio viewer.

The RStudio interface includes a top bar with tabs like 'Activities', 'RStudio', 'File', 'Edit', etc., and a bottom dock with various application icons.

Figure 2 Creating and adding fields to a table using R.

- The above screenshot shows the steps to create a table and add fields to a table using R inside RStudio.
 - What the code numbered as 1 on the screenshot does is connecting to a database named ‘bigdata’ using the provided credentials. The code designated as 2 creates a table named ‘users’ with the provided attributes. Number 3 adds a ‘PRIMARY KEY’ ‘id’ to table ‘users’. Number 4 inserts data in to table ‘users’. And finally, number 5 assigns the result from the select statement to a variable user in order to view it inside RStudio. The result from executing code number 5 is shown in the figure below.

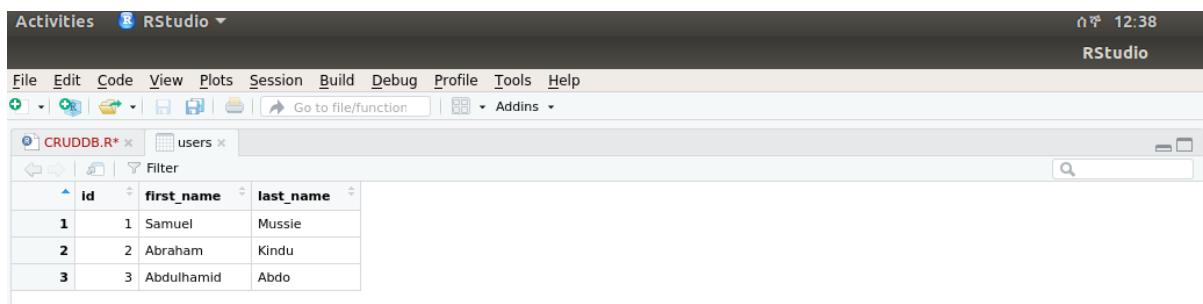


Figure 3 Executing View(users).

- *Executing update operation.*

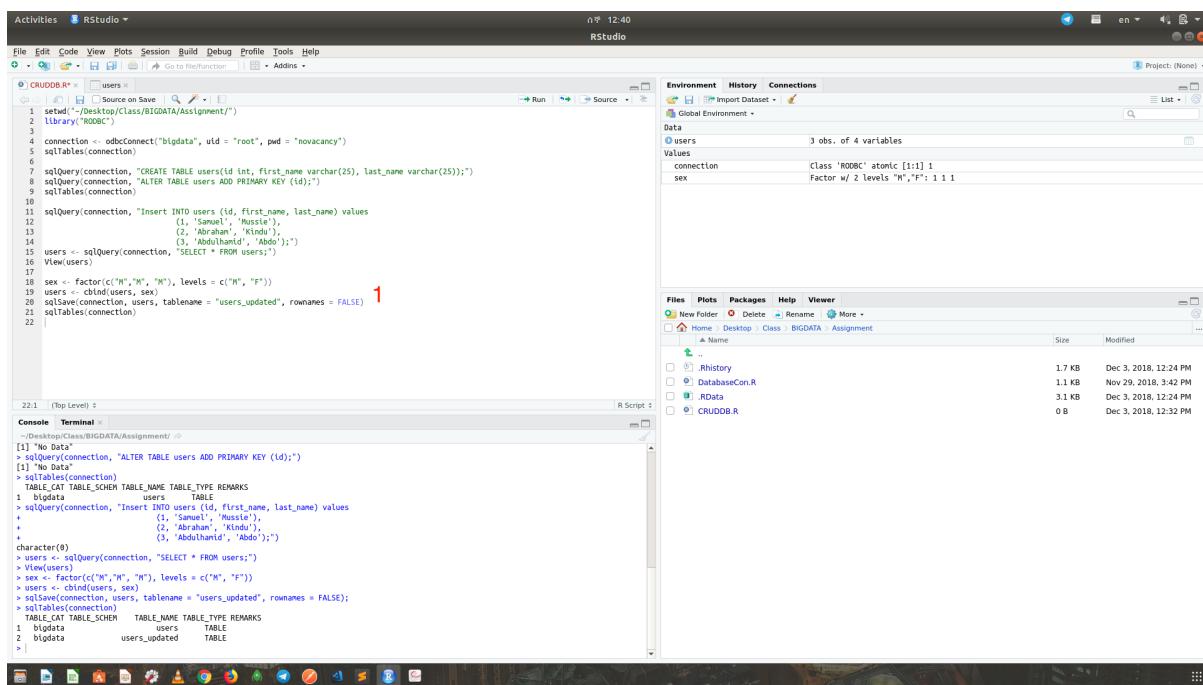


Figure 4 update operation.

- The codes marked as I will factor and add a 'sex' attribute. Then cbind is called to bind 'sex' attribute and then we add it as a column in 'user_updated' table. The contents of the table are shown in the figure below. This showcases question (d) and the figure below showcases question (e). As the list is too small we will be viewing the whole data.

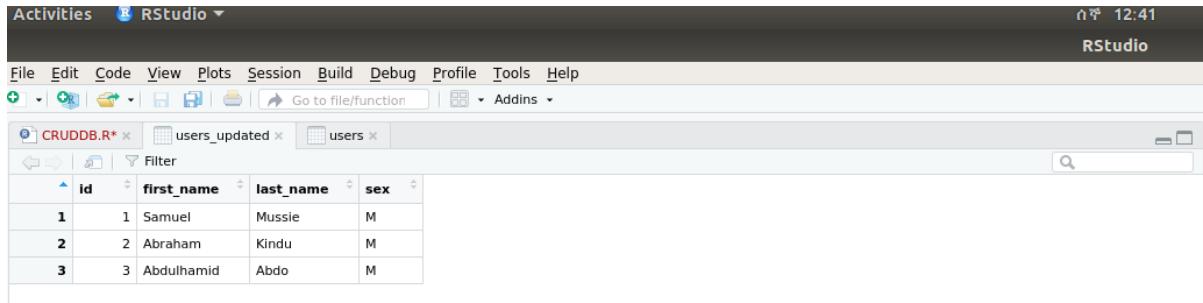


Figure 5 Viewing the updated table.

- In similar manners, we create table ‘addresses’ and ‘grades’.
 - Creating table addresses.

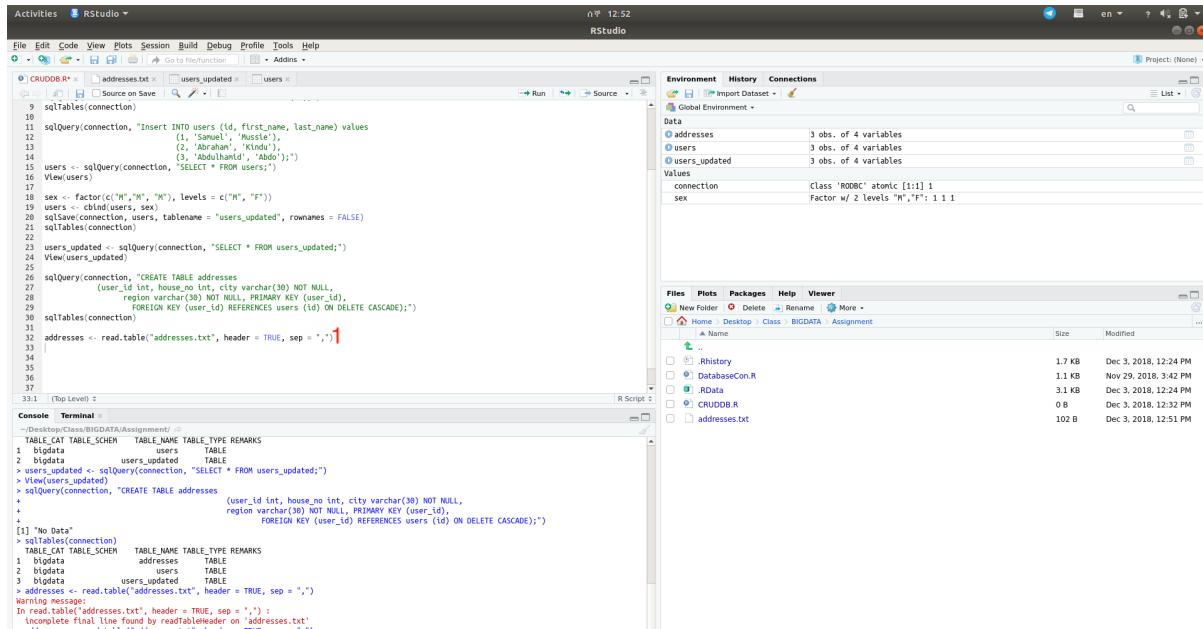


Figure 6 Creating and inserting data to table addresses.

- Code number 1 reads addresses.txt and adds them to table address. The contents of addresses.txt is shown in the figure below.

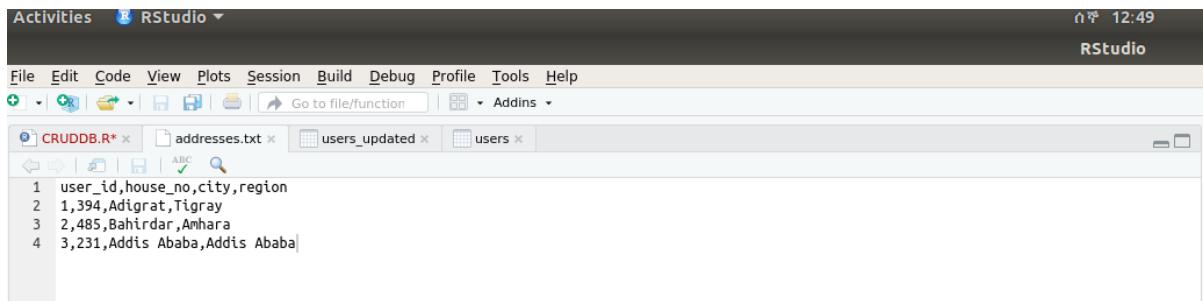


Figure 7 Addresses.txt

- *Creating table ‘grades’.*

The screenshot shows an RStudio interface with several windows open:

- File Explorer**: Shows the project structure with files like `CRUDDR.R`, `db_addresses`, `addresses`, `addresses.txt`, `users_updated`, and `users`.
- Code Editor**: Displays the `CRUDDR.R` script. A red arrow points from the `grades` table definition (lines 41-45) to the `sqldf` documentation in the Help Viewer.
- Environment**: Shows the global environment with objects like `addresses`, `db_addresses`, `users`, and `users_updated`.
- Help Viewer**: The `sqldf` documentation is open, showing functions for saving and updating data frames to ODBC databases.

Figure 8 Creating table grades.

- *Code numbered as 1 creates a table named grades with the provided attributes. We can see that, after the execution of sqltables(connection), a new table named grades is added as shown in the console.*
 - *Adding and updating table grades.*

The screenshot shows the RStudio interface with several panes open:

- Code Editor:** Displays R code for creating a table named "grades" and inserting data from "grades.csv". The code uses `sqldf` and `odbc` packages. A red '1' is placed next to the line `student_grades <- read.csv("grades.csv")`.
- Environment:** Shows the global environment with objects like `addresses`, `db_addresses`, `student_grades`, `users`, and `users_updated`.
- Help:** The right pane displays the documentation for the `sqldf` package, specifically the `update` function. It includes sections for Description, Usage, Arguments, and Examples. A red '2' is placed next to the 'Description' section.

Figure 9 adding and updating table grades.

- Code numbered as 1 reads grades from file `grades.csv` and assigns it to a variable named `student_grades`. Codes designated as 2 filters student grades and assigns letter grades based on grade values.

- Contents of `student_grades` before adding letter grades are shown in the figure below.

	user_id	course_name	grade
1	1	Big data	86
2	1	Robotics	91
3	1	Database	76
4	2	Database	87
5	3	Database	64
6	3	Robotics	74
7	2	Big data	69
8	2	Robotics	66
9	3	Big data	90

Figure 10 Student_grades

- The content of `grades` table is shown in the figure below.

	id	user_id	course_name	course_grade
1	1	1	Big data	A
2	2	1	Robotics	A+
3	3	1	Database	B+
4	4	2	Database	A
5	5	3	Database	D
6	6	3	Robotics	B
7	7	2	Big data	B
8	8	2	Robotics	D
9	9	3	Big data	A+

Figure 11 Grades with letter grades

- *Update and delete operations*

Figure 12 Update and delete operations.

- The code numbered as 1 updates tables grades by looking for course_grades with values 'D' and setting them to value 'C'. The code numbered as 2 deletes a user with id 0f 1. Updated table and deleted table are shown in the figure below respectively.

The figure displays two RStudio sessions side-by-side, each showing a data frame named 'grades'. The left session shows a data frame with 9 rows and 4 columns: id, user_id, course_name, and course_grade. The right session shows a data frame with 9 rows and 4 columns: id, user_id, course_name, and course_grade. A red arrow points from the 5th row of the left frame to the 5th row of the right frame, and another red arrow points from the 8th row of the left frame to the 8th row of the right frame.

	id	user_id	course_name	course_grade
1	1	1	Big data	A
2	2	1	Robotics	A+
3	3	1	Database	B+
4	4	2	Database	A
5	5	3	Database	D
6	6	3	Robotics	B
7	7	2	Big data	B
8	8	2	Robotics	D
9	9	3	Big data	A+

	id	user_id	course_name	course_grade
1	1	1	Big data	A
2	2	1	Robotics	A+
3	3	1	Database	B+
4	4	2	Database	A
5	5	3	Database	C
6	6	3	Robotics	B
7	7	2	Big data	B
8	8	2	Robotics	C
9	9	3	Big data	A+

Figure 13 Updated grades table values

The screenshot shows the RStudio interface with the following details:

- Top Bar:** Activities RStudio ▾ 14:01
- Menu Bar:** File Edit Code View Plots Session Build Debug Profile Tools Help
- Toolbar:** Includes icons for New, Open, Save, Print, and Go to file/function.
- File List:** JDDB.R* db_grades grades.csv db_addresses addresses addresses.txt users_updated users
- Data View:** A data frame titled "grades" with the following content:

	id	first_name	last_name
1	2	Abraham	Kindu
2	3	Abdulhamid	Abdo

Figure 14 Deleted User with id 1

- Showing cascading delete from the grades table too. No grade is shown with user id of 1 as user with an id if 1 is deleted his grades will also be automatically deleted.

The screenshot shows the RStudio interface with a database connection named 'CRUDDB' open. The 'Environment' tab displays various data frames: 'addresses', 'db_addresses', 'db_grades', 'grades', 'student_grades', 'users', and 'users_updated'. The 'Console' tab shows the following R code and its execution:

```

library(DBI)
library(odbc)
library(dplyr)

# Establishing a connection to the ODBC database
connection = odbc::odbc_connect("CRUDDB")

# Displaying the structure of the db_grades table
db_grades %>% 
  select(-id) %>%
  summarise_all(numeric)

# Executing a SQL query to select all rows from the grades table
db_grades <- sqlQuery(connection, "SELECT * FROM grades;")

# Summarizing the db_grades data frame
summary(db_grades)

# Viewing the db_grades data frame
View(db_grades)

```

The 'Console' output shows the following results:

id	user_id	course_name	course_grade
1	4	Database	A
2	5	Database	C
3	6	Robotics	B
4	7	Big data	B
5	8	Robotics	C
6	9	Big data	A+

The 'Environment' pane shows the following data frame structures:

- addresses: 3 obs. of 4 variables
- db_addresses: 2 obs. of 4 variables
- db_grades: 6 obs. of 4 variables
- grades: 9 obs. of 4 variables
- student_grades: 9 obs. of 4 variables
- users: 2 obs. of 3 variables
- users_updated: 3 obs. of 4 variables

The 'Values' pane shows the following factor levels:

- course_grade: 6 levels: "D" < "B" < "A" < ... : 5 6 3 5 1 2 2 1 6
- letter_grades: 9 levels: "A" < "A+" < "B" < "B+" < "C" < "C+" < "D" < "D+" < "F"
- sex: 2 levels: "M", "F": 1 1 1

The 'Description' pane indicates: 'R: Write a Data Frame to a Table in an ODBC Database'.

The 'Usage' section of the help page for `sqlUpdate` is visible:

```

sqlSave(channel, dat, tablename = NULL, append = FALSE,
        rownames = TRUE, colnames = FALSE, verbose = FALSE,
        safer = TRUE, addPK = FALSE, typeInfo, varTypes,
        fast = TRUE, test = FALSE, nastring = NULL)

sqlUpdate(channel, dat, tablename = NULL, index = NULL,
          verbose = FALSE, test = FALSE, nastring = NULL,
          fast = TRUE)

```

The 'Arguments' section of the help page for `sqlSave` is visible:

Argument	Description
channel	connection handle returned by <code>odbcConnect</code> .
dat	a data frame.
tablename	character: a database table name accessible from the connected DSN. If missing, the name of dat.
index	character: Name(s) of index column(s) to be used.
append	logical: Should data be appended to an existing table?
rownames	either logical or character. If logical, save the row names as the first column rownames in the table? If character, the column name under which to save the rownames.
colnames	logical: save column names as the first row of table?

Figure 15 Demonstrating cascading delete