Review for DATA/COSC 301/DATA 501 Exam

Proposed Format

Time limit: 180 minutes
Total marks: 100

- $\sim$ 30 marks for 30 multiple choice questions (1 mark each)
- $\sim$ 20 marks for 7 short answer questions
- $\sim$ 50 marks for 7 long answer questions

Previous Exam Breakdown

Last year, the exam has covered 36% midterm 1 material, 27% midterm 2 material, leaving 37% of the material unique to final. I am not bound to the exact percentages, but they should give you a rough idea on how to study. There is some variation in this distribution each semester, but there will be roughly a 1:1:1 coverage of Midterm 1, Midterm 2 and post Midterm 2 material.

Table 1: MC = Multiple Choice (1 mark each), SA = Short Answer, LA = Long Answer

Percentage	Topic	Distribution
2%	Data representation	2 MC
11%	Excel	1 MC, 1 LA (10)
1%	Excel VBA	1 MC
16%	Databases	2 MC, 1 SA (4), 1 LA (10)
6%	Command line[1]	1 MC, 1 LA (5)
27%	Python	12 MC, 3 LA (3x5)
21%	R	7 MC, 1 SA, 1 LA (10)
7%	GIS[1]	1 MC, 2 SA
5%	Visualization	3 MC, 1 SA
4%	Open data	2 SA

[1] Since we did not cover Command Line and GIS, these percentages will be redistributed across the remaining topics

1 List of Topics

Table 2: Relevance of topics as it pertains to the final exam

***	Extremely important
**	Assignment question or major topic
*	Important topic which probably should be tested
-	(no stars) topic covered but of minor importance
~~strikethrough~~	Will not be tested on the exam

Introduction

* what is data analysis? what does a data analyst do?
* importance of data analytics

Data Representation

* Define: computer, software, memory, data, memory size/data size, cloud
* Explain “Big Data" and describe data growth in the coming years
** How data is measured: bits vs bytes vs KB, GB, ...
* Compare and contrast: digital versus analog
** How integers, doubles, and strings are encoded
** Difference between unsigned binary, float (32 bit), double (64 bit)
** Convert integer into unsigned binary
- Convert real number into float
* Why ASCII table is required for character encoding
* Explain why Unicode is used in certain situations instead of ASCII
** Explain the role of metadata for interpreting data
* Define: file, file encoding, text file, binary file
- Encode using Hexadecimal, the NATO broadcast alphabet
- Discuss the time-versus-space tradeoff

Excel

* Explain what a spreadsheet is and its usefulness

** Excel notation: ribbon, worksheet, workbook
** spreadsheet cell addressing (eg. range notation using :)
- selecting cells in a spreadsheet with your mouse
- filling, hiding
** Define and explain: formula, function, argument, concatenation
** Using functions, eg. concatenate, lookup, index, if, aggregate functions
*** compare absolute vs. relative addresses ; use absolute/relative addresses
** use conditional formatting, format painter
** data and type formats
** Use sorting and filtering (with numbers or characters)
** Create/edit/identify different charts
* Create and edit charts and use chart features: trendlines, sparklines
* Explain the usefulness of: what-if scenarios, goal seek, solver
*** Use and create pivot tables and charts (understand the ROW, COLUMN, FILTER, and VALUES panels)
** Evaluate and create conditions.
** Use IF() to make decisions
* Excel add-ins: define, how to install (eg. Solver. Data Analysis)
- Linear regression

Excel VBA

*** Explain/understand how to create and use macros and macro recorder
** Excel file extensions: mainly know the difference between .xlsx and .xlsm
* Saving macros: Personal workbooks (.xlsb) vs regular .xlsm
** Explain the security issues with macros and how to handle them
** Visual Basic Editor: basic definition and features
    * How to use the immediate window, eg `?/PRINT/DEBUG.PRINT <command>`
    - VBA modules
    - Object browser
* Be able to read/manipulate macro/VBA code (don't need to create from scratch but you should know the general syntax)
    - Object-oriented definitions (object, class, property, method)
    * The hierarchy of objects eg. Application → Workbook → Worksheet

* Collections and indexing, eg. `Worksheets("macro")`, `Worksheets(2)` (index starting at 1)

** Create and use Excel variables

* Explain how a collection is different from a typical variable

* eg. `Range` for selecting cells

** Explain how to create user-defined functions and use them in formulas

* Subroutines vs user-defined functions.

Relational Databases

** Define: database (*data*), database system (*software*), schema, metadata

*** Define: relation (table), attribute (field/column name), tuple (a single record/row), domain (data type for a column), degree (number of columns/attributes), cardinality (number of rows/tuples)

* SQL properties: reserved words, case-insensitive, free-format

- GUI commands in Microsoft Access and LibreOfficeBase (i.e Design View)

*** Write queries using SQL commands

N.B Some commands differed slightly between LibreOffice Base and Access; both will be accepted.

*** Be able to create a table using CREATE TABLE

*** Know your field types (eg. `VARCHAR` vs `CHAR`)

** Explain what a primary key is and what it is used for and how to assign one

* Use DROP TABLE to delete a table and its data

** Use INSERT/UPDATE/DELETE to add/update/delete rows of a table

* ALTER command for adding new columns

*** Projection operation using SELECT

** DISTINCT to remove duplicates

*** Selection operation using WHERE

*** comparison operators (`<, >, =, !=, <=, >=`),

*** logical operators (`AND, OR, NOT`).

- `IS NULL`

*** JOINS (inner, left, right, outter – know the difference)

** Sort rows using ORDER BY (ASC/DESC for ascending/descending, resp.)
** Use LIMIT/TOP to keep only the first (top) N rows
*** Use GROUP BY and aggregation functions (eg. `COUNT`, `MAX`) for calculating summary data
** HAVING for filtering after GROUPBY.
*** Statement Written/Execution Order

07Python

- Explain what is Python and note the difference between Python 2 and 3 (I will be marking using Python 3 syntax)
* Define: algorithm, program, language,
** Follow Python basic syntax rules including indentation/comments/colons
- Jupyter Notebook
*** Using `print` statements
    - Use new-style string formatting, (feel free to use, but you don't need to) eg. `print("Total score for {} is {}".format(name, score))`
* Perform math expressions and understand operator precedence
*** Print formatting
*** Use strings, character indexing, string functions, slicing
** Commonly used string functions/methods: split, substr, concatenation (+), escape character
*** Define and use variables and assignment
** Python naming rules
* Python data types (see using `type` function)
*** Python indexing `[]` starts at 0, -1 right to left indexing
** Using functions and methods, eg. `len()`, `.append()`, `str()`, `int()`, `.split()`
* Deleting objects `del`, `.remove`
** How to import and use modules
- Use Python datetime and clock functions
*** Read input from standard input (keyboard)
*** Create comparisons and use them for decisions with `if`
** Combine conditions with `and`, `or`, `not`
** Order of operations for logical operators, eg `and` before `or`
*** `if/elif/else` statements

*** Looping with `for` and `while` (eg. using `for` loops with Python lists, strings., etc.)

*** `range(start (inclusive), end (exclusive), step (optional)`

*** Create and use lists, list functions (eg. `.sort`), and list slicing, traversing in `for` loop

* Python collections: tuples, sets

* Differences between lists, tuples, sets and dictionaries

*** Create and use dictionaries

- Advanced: list comprehensions

*** Create and use Python functions

- Difference between Python functions and procedures

** Differences between functions and methods

- lambda functions

- Use built-in functions in math library

- Create random numbers

- Advanced: passing functions, lambda functions

08Python

*** Open, read, write, append to files

** Closing files (either in a `with` clause or using `fileobj.close()`)

*** Process CSV files: you may either use base Python *or* use the csv module (but this module makes it easier)

** Define: exception, exception handling

*** Use try-except statement to handle exceptions and understand how each of try, except, else, finally blocks are used

- Define: IPv4/IPv6 address, domain, domain name, URL

- Read URLs using urllib.request.

- ~~Use Biopython module to retrieve NCBI data and perform BLAST~~

* Build charts using matplotlib (eg alter existing code to produce some desired affect)

- Perform linear regression and k-means clustering using SciPy

- Connect to and query the MySQL database using Python

- NumPy arrays

* Write simple Map-Reduce programs

Open Data

*** Key concepts/take home message from Dr. Jason Pither's guest lecture

*** Define open data and explain the motivations for making data "open".

* Terminology (eg. reproducibility)

* Publication bias

- Power/ any technical details

* Ideal workflow

** pre-registration of research plan

* Tools for helping with Open Science

R

** Understand purpose and usefulness of R

** Types of data: qualitative, quantitative

*** Describe data using numerical summaries (measure of centre/spread, fivenum)

*** Define and calculate: mean, median, variance, standard deviation, range

** Define: quantile, quartile, interquartile range, five number summary

** Explain what a working directory is and how to set it

*** R syntax, for example: `{}` for blocking lines of code, case sensitive, `<-`/`=` for assignment character, ...

* Perform matrix addition, subtraction, and multiplication

- Install and use RStudio (no RStudio specific questions)

** Write small programs/commands in R that may use variables, conditions, logicals, loops, and functions

** Read in data sets from files (eg. using `read.csv` or `read.table`)

** Use `head` and `tail`

** Create and use data structures: vectors, matrices, lists

** Indexing vectors/matrices/arrays/lists, *remember that indexing starts from 1 in R!*

** Use data frames/factors for data analysis

* Create graphs/visualizations: frequency table, bar chart, histogram, boxplot, using ggplot2 or base (eg. alter code, map code to figure, expected output)

** List the assumptions of a t-test

** Compute and read output of linear models using R

- Using SQLish functions (eg. `subset`) may be helpful when writing code, but you won't be tested on the explicitly

* Handling missing data

- `repeat`, `next`, `break`
- `apply()` functions
- k-means

*** Perform hypothesis testing and read output

*** Stating the null/alternative hypothesis

* identifying test statistic

*** Decision and conclusion based on p-values

*** Choosing the most appropriate test (one-sample, independent two-group, paired two-group)

- Explain the purpose of confidence intervals

*** linear regression

*** State the fitted regression line

* residuals

* State the R-squared values

*** Predict y for a given x

~~GIS~~

* ~~Provide examples where a GIS is used~~

⋮

- ~~Use Python to connect to Google Maps API~~

~~Command Line~~

** ~~Define command line and list some of its uses~~

⋮

** ~~Be able to connect to another machine using SSH~~

Visualization

* List different types of visualizations available in Excel, Python, R ~~GIS~~

** Create basic plots using Excel, Python, and R