

Teaching Finance

with KNIME Analytics Platform



Table of Contents

Introduction	3
Just KNIME It! Challenges.....	4
Days with Price Changes.....	4
Date to Fiscal Year	4
Retiring Early	5
Implementing Custom Time Alignment	5
Comparing Exchange Rates	6
Comparing Monthly Sales	6
Calculating YTD and MTD	7
Dashboard to Update Actuarial Information.....	7
Trends in Stock Prices	8

Introduction

Finance departments are at the heart of each company. Their activities include planning, forecasting, budgeting, and analysis to support major business decisions and analyze the overall financial health of the company, as well as accounting and treasury tasks. All those activities require precision, correctness, repeatability, and speed. To obtain reliable and fast procedures, many companies have started implementing them with workflows in tools, like KNIME Analytics Platform, that provide the same level of reliability as programming languages but can be adopted by non-programmers.

We have compiled this document to provide some inspiration and starters for how KNIME Analytics Platform can support the teaching of (digital) finance topics to financial professionals in various stages of their career. That is, you will find suitable materials for undergraduate as well as graduate and continuing education courses.

With this package, you have received **presentations covering various use cases** that you can use in your finance teaching, in addition to course materials available through the [KNIME Educators Alliance](#):

- *2024-08-27_Data_Analytics_and_AI_for_Finance.pptx*
 - How to leverage KNIME to automate finance tasks
 - How to make use of AI techniques to simplify processes
 - Delve into two examples: Bank Reconciliation and Fraud Detection
 - [Recording](#)
- *Use Cases.pptx*
 - Reusable collection of 200 slides covering 10+ use cases from the KNIME for Finance collection

If you'd like to take a hands-on approach, you will find all the workflows for the mentioned use cases in a dedicated [KNIME for Finance](#) Collection Page. These workflows have been built to be **ready-to-use solutions** for finance tasks both on the job as well as in educational settings. You will find links to the blog articles for each item in the collection: They all come with a short video, which we have also compiled into a [KNIME for Finance YouTube Playlist](#).

The remainder of this document details **hands-on challenges** that can be assigned as exercises in classes to make the learning more interactive. These challenges were originally published as [Just KNIME It!](#) challenges, for which participants had one week to tackle them. This makes them great choices for lab exercises to enhance your learners' KNIME skills with data they can easily relate to. The bigger picture is often secondary for these challenges since they focus more on technical upskilling. You can find more challenges as well as detailed descriptions of the solutions in the Just KNIME It! [Season 1](#) and [Season 2](#) booklets.

Just KNIME It! Challenges

Days with Price Changes

Level: Easy

Description: You are using KNIME to monitor the daily price of a product online. After using the Line Plot node to visualize the daily prices you have already gathered, you notice that they are often constant for a certain number of days before changing again. You want to create a new column in the price data you have at hand, named "Change", such that its value is 1 if a daily price changed with respect to the previous day, or 0 if it remained unchanged. For the first daily price in the data, the "Change" value should be 1.

As an example, if the initial daily prices look like:

Date	Price
2015-01-01	10
2015-01-02	10
2015-01-03	10

You should end up with data in the following format:

Date	Price	Change
2015-01-01	10	1
2015-01-02	10	0
2015-01-03	10	1

Dataset: [Daily Prices in the KNIME Community Hub](#)

Solution Summary: To create column "Change" according to the challenge's description, we first created a temporary column, named "Price(-1)", holding the price value for the previous day (that is, a column just like "Price", but with a 1-day lag). Next, we compared the lagged and current prices to determine whether there was a change in value or not, leading to the creation of column "Change". Finally, we remove temporary column "Price(-1)" from the final dataset.

[See our solution in the KNIME Community Hub](#)

Date to Fiscal Year

Level: Medium

Description: In the accounting firm you work for, you are given contracts which were executed on different dates. Your goal is to create a method that can label each of these contracts with their corresponding fiscal year. **Note:** A fiscal year is a period of 12 months that is used in government accounting, usually for budget purposes and financial reporting. Its definition varies from country to country, so your solution should be flexible and include flow variables for both the start and the end dates of the fiscal year. As an example, the federal fiscal year in the United States is a 12-month period that begins on October 1st and ends on September 30th of the following year.

Dataset: [Labels for fiscal years and contract dates in the KNIME Community Hub](#)

Solution Summary: We started by reading the contract dates and the lookup tables with the defined fiscal years. Next, for each fiscal year, we selected those contracts whose dates fell within it and labeled them accordingly. The process of identifying which contracts fall within a given fiscal year was made flexible with the use of flow variables and the Date&Time-based Row Filter node. After all contract dates were labeled with their fiscal years, we sorted them by date to facilitate the understanding of the data.

[See our solution in the KNIME Community Hub](#)

Retiring Early

Level: Easy

Description: Your coworker turns to you and says that she is going to retire. You laugh because she is 30 years old. She is serious. To understand how she got to this decision, you will create a KNIME component named "Financial Tracker_YOURNAME" (replace YOURNAME with your name). The component should use widgets to get the following input:

1. a person's monthly expenditure amount
2. their target age to retire

The output of the component should be how much money they need to have in order to retire at the target age. For simplicity, use this formula in your component:

$$\text{amount_to_retire} = (100 - \text{target_age}) * \text{monthly_expenditure_amount} * 12$$

Use your component to figure out if 2,000,000 dollars is enough for your coworker to retire, given that she spends 4,000 dollars per month. To keep this challenge simple, do not consider inflation, compounding interest, or part-time work in retirement.

Are you interested in how we came up with this formula? Check the [Trinity study](#) out. In this study, participants needed roughly 25 times whatever they spent yearly to survive for 30 years with a 95% success rate.

Solution Summary: To solve this challenge, we created a component in which we first use widgets to capture monthly expenses and target age for retirement. Next, we calculated the amount of money required to retire at the target age, given the monthly expense, following the formula given in the challenge. We used another widget to show the calculated value and also added a "Recalculate" option for users to explore different retirement setups.

[See our solution in the KNIME Community Hub](#)

Implementing Custom Time Alignment

Level: Medium

Description: KNIME has just released a new textbook on time series: to celebrate it, we will do a little time series analysis in this challenge.

For this challenge, you will perform time alignment on data that contains two types of gaps: regular gaps by the nature of the data, and irregular gaps. For example, daily stock market data

regularly skips Saturdays and Sundays, and irregularly skips some other weekdays due to public holidays. Your concrete task for this challenge is to introduce the missing timestamps that correspond to weekdays into the given data while omitting those for weekends. The data contains the daily exchange rates of US dollar vs Swiss franc from 1980 to 1998. **Hint:** Check out our [verified components for time series analysis](#).

Data: [Daily Exchange Rates Data in the KNIME Community Hub](#)

Solution Summary: In our solution, we started by extracting the year (week-based) and week associated with each date in the dataset. Next, we grouped the data by week and, for each group, we inserted sequential timestamps for those weekdays that were missing. The final dataset thus included (1) the original data on a daily granularity, and (2) missing entries for those weekdays that were not in the original data.

[See our Solution in the KNIME Community Hub](#)

Comparing Exchange Rates

Level: Easy

Description: You are doing research on trading and different currencies and one of your tasks is to compare the daily exchange rates of US dollar vs 8 other currencies from 1980 to 1998. The data contains some missing values due to public holidays on weekdays, which you should handle first using a strategy of your choice. After that, you should calculate the correlations between the exchange rates of the currencies and visualize them. Which currencies had the highest positive/negative correlation? Which currencies were not correlated, or had a low correlation?

Dataset: [Exchange Rates Data on KNIME Community Hub](#)

Solution Summary: After reading the dataset with daily exchange rates, we used the simple strategy of replacing missing values with previous dates'. Next, we computed the linear correlations across all currency pairs using Pearson's coefficient, and then visualized them with a heatmap. The highest positive correlations involved the Swiss, German, and Dutch currencies. The highest negative correlation was between the Australian and Japanese currencies. The correlations closest to zero were between the Canadian currency and the other currencies.

[See our Solution in the KNIME Community Hub](#)

Comparing Monthly Sales

Level: Medium

Description: The sales manager at *ABC Grocery Stores* is looking for a way to compare the number of sales of a selected month with those of a prior month. You are tasked to create a component in KNIME that carries out this task, with a visualization tackling the comparisons. If there are no prior months then only data for the selected month should be shown. If a prior month can be picked for comparison, it should not be more than 12 months before the selected.

Dataset: [Monthly Sales Data on KNIME Community Hub](#)

Solution Summary: To solve this challenge, we built a component that allows users to (1) select the month against which they want to compare sales numbers, and (2) a second previous month based on a gap. As an example, users can pick June 2020 for (1) and, by entering a gap of -3,

also pick March 2020 as (2) for comparisons. The component also contains a bar chart to compare sales numbers for both selected months and a refresh button that allows users to vary their selections and resulting plots.

[See our Solution in the KNIME Community Hub](#)

Calculating YTD and MTD

Level: Easy

Description: Restaurant Yummy records its sales values on a daily basis. The data contains two columns: *Date* and *Sales*. Year-to-Date (YTD) Sales and Month-to-Date (MTD) Sales are important metrics to track the revenue of the business. But what exactly are YTD and MTD values? The YTD value allows you to calculate the metric (e.g., sum of sales) for the current year, while the MTD value denotes a metric value for the current month. You are asked to build a KNIME workflow that takes this data as input and adds YTD and MTD values across each record.

Dataset: [Sales Data in the KNIME Hub](#)

Solution Summary: To tackle this challenge, we group the data by month and, using lags based on the previous months' values, calculate the MTD. We also use moving aggregation to go over the sums of sales to calculate the YTD.

[See our Solution in the KNIME Community Hub](#)

Dashboard to Update Actuarial Information

Level: Medium

Description: You are a freelance data scientist and are asked to help an actuarial agency. To compute premiums for life insurance, actuaries need to find the age-dependent information about customers from an Actuarial Life table containing the probability of dying within 1 year and life expectancy. Your task is to make this process faster: create a dashboard in which an actuary specifies the age of the customer and this action updates probabilities and life expectancy for both genders. Feel free to use either tables or graphical representations of data. **Hint:** To combine values of the first 3 rows to use as column headers, refer to [this workflow](#) from the Community Hub. **Hint 2:** The Integer Widget does not refresh the interactive view of the component. Consider adding the Refresh Button Widget.

Dataset: [Actuarial Life Table in the KNIME Community Hub](#)

Solution Summary: To tackle this challenge, we start by reading the Actuarial Life table, processing its header values, and then combining them. Next, we filter out columns that are not relevant to the problem, and then create a dashboard in which users can check the average number of remaining years a population has depending on their age (which they can choose), along with their individual probabilities of dying within a year.

[See our Solution in the KNIME Community Hub](#)

Trends in Stock Prices

Level: Easy

Description: You work in finance and one of your clients wants to understand the value of different company stocks over time. Given a dataset of stock prices, you decide to use **simple moving averages** (window length = 20) to tackle this task. What companies have an upward trend for the most recent data? And what companies have a downward trend?

Dataset: [Stock Data in the KNIME Community Hub](#)

Solution Summary: We propose two different solutions to this challenge. The simplest one involves manually filtering the data for a specific company, calculating its moving average, and then visualizing it with a line plot. The second one relies on a simple data app: a company is selected from a dropdown box and its stock prices are selected, a moving average is computed, and the final points are plotted as a line plot.

[See our Solution in KNIME Community Hub](#)

KNIME AG
Talacker 50
8001 Zurich, Switzerland

www.knime.com
info@knime.com

The KNIME® trademark and logo and OPEN FOR INNOVATION® trademark are used by KNIME AG under license from KNIME GmbH, and are registered in the United States. KNIME® is also registered in Germany