Data Sleek Data Warehouse Project - QuickBooks

UCLA Anderson MSBA 405
2023 Database Management Final Project

Group 5
Justin Braun
Raven Chen
Marilyn Cheng
Siying Li
Khang Nguyen
Abinash Panda
Vishaal Prasad
Rohan Singh

Table of Contents

Executive Summary

Data Sleek (DS) is a data consulting and management company that works with clients across the world to develop data-based strategies and deliver tailored solutions in data architecture, data engineering, data science, and data warehousing. Located in Los Angeles, California, DS helps businesses unlock the full potential of their data and maximize investments. QuickBooks is an accounting software that allows DS to manage finances and streamline their accounting processes. It can be integrated with many other applications and is used to track income, expenses and inventories, create and send invoices, and generate financial reports.

Project Statement

The purpose of our project is to partner with Data Sleek to analyze the data in their data warehouse using Snowflake, design KPIs (Key Performance Indicators) that are easy to analyze, and produce interactive dashboards in Tableau. The scope of our project is to identify relevant data sources from QuickBooks in the DS data warehouse, analyze the data using SQL queries, design and execute KPIs that align with the business objectives, and create a dynamic dashboard that showcases these KPIs clearly and concisely. The Tableau dashboard will display real-time data and can be customized to meet the needs of different departments and stakeholders in DS.

Data Dictionary

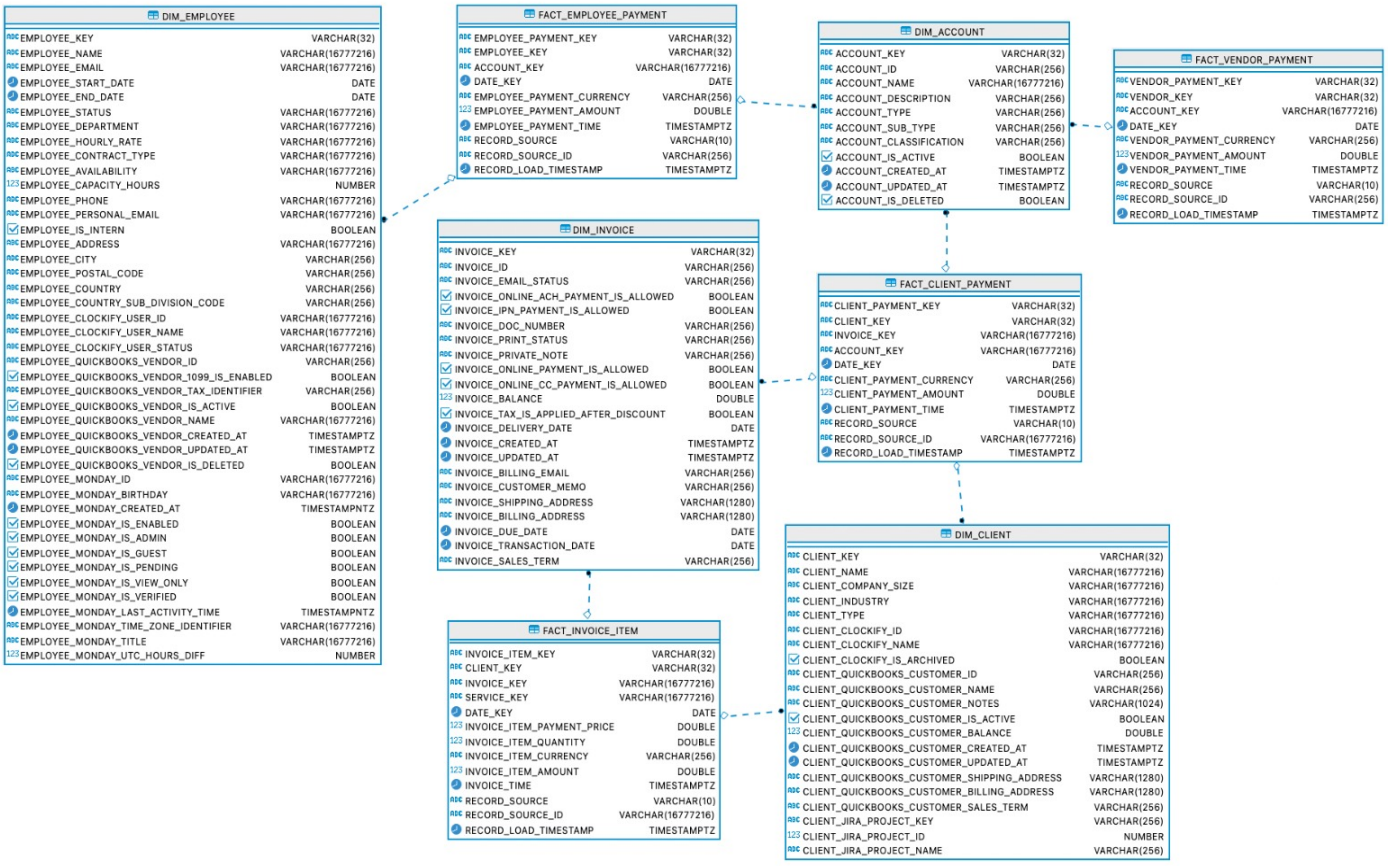
The below table is a brief display of the data elements, attributes, and entities present in the data we have extracted from Data Sleek's data warehouse for the purpose of visualizing and analyzing. This includes the column names, tables in which they are located, their respective data types, and a brief description of each column.

Column Name	Table	Type	Description
client_name	dim_client	VARCHAR(16777216)	Name of the Data Sleek client
employee_capacity_hours	dim_employee	NUMBER(38,0)	Number of hours the Data Sleek employee is capable of providing service
invoice_due_date	dim_invoice	DATE	Due date of payment for a client invoice
service_name	dim_service	VARCHAR(16777216)	Name of the Data Sleek service provided
client_payment_amount	fact_client_payment	FLOAT	Amount paid by the client
date_key	fact_employee_payment	DATE	Unique identifier of a date

employee_payment_amount	fact_employee_payment	FLOAT	Amount paid to the Data Sleek employee
invoice_item_amount	fact_invoice_item	FLOAT	Total amount for the invoice
invoice_item_key	fact_invoice_item	VARCHAR(32)	Unique identifier of an item in the invoice
invoice_item_payment_price	fact_invoice_item	FLOAT	Unit price of an item in the invoice
invoice_item_quantity	fact_invoice_item	FLOAT	Quantity of an item in the invoice
invoice_key	fact_invoice_item	VARCHAR(16777216)	Unique identifier of the invoice
vendor_payment_amount	fact_vendor_payment	FLOAT	Amount paid by the vendor

ERD (Entity Relationship Diagram)

The below diagram showcases the entity relationship diagram of how all entities within the underlying Quickbooks fact and dimension tables relate to each other.



Data Literacy

The QuickBooks dataset consists of accounting information of two main categories (transactions and entity accounts) between Data Sleek and three entities, namely clients, vendors, and employees. The data is stored in multiple dimension and fact tables, with each table serving a unique purpose.

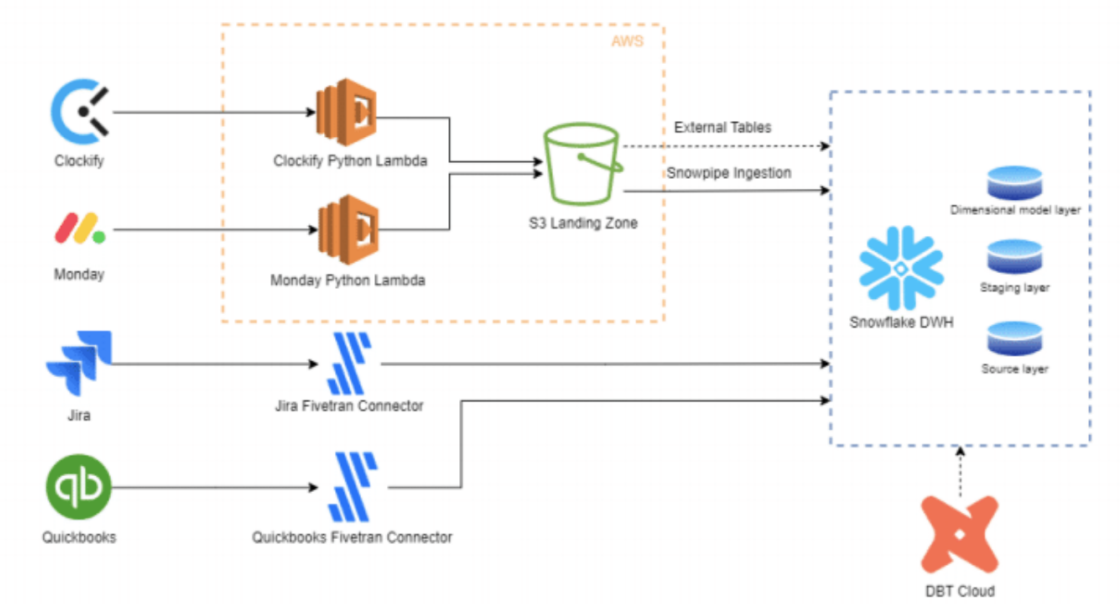
Transactional data includes information about invoices, purchases, and client payments, with each line item representing an accounting transaction. It is worth noting that invoice data is a little more unique, as there are two tables relating to invoice. DIM_INVOICE is a table with individual invoices as its grain, while FACT_INVOICE_ITEM has each individual service billed as its grain.

For accounts data, the tables contain information about each entity. Edits to the accounts are updated directly, overwriting existing data. The modification date is stored as a column in the table (xxx_updated_at), and each account ID only has one record in the table.

One of the essential aspects of understanding this dataset is its volume of data. The transactional files, such as FACT_INVOICE_ITEM, DIM_PURCHASE, and FACT_CLIENT_PAYMENT, have a higher volume of data, with thousands of rows. In contrast, the account files' volume of data is significantly smaller as seen below. The following table showcases the fact and dimension tables available to us in the dataset.

name	rows	bytes
FACT_INVOICE_ITEM	4,486	276,992
DIM_PURCHASE	2,272	114,176
FACT_CLIENT_PAYMENT	1,029	97,792
FACT_VENDOR_PAYMENT	1,406	95,232
DIM_INVOICE	1,078	92,160
DIM_PAYMENT	1,029	49,152
FACT_EMPLOYEE_PAYMENT	527	49,152
DIM_EMPLOYEE	98	33,792
DIM_CLIENT	201	32,768
DIM_VENDOR	274	24,064
DIM_ACCOUNT	69	11,264
DIM_SERVICE	57	8,192

Data Processing and Architecture



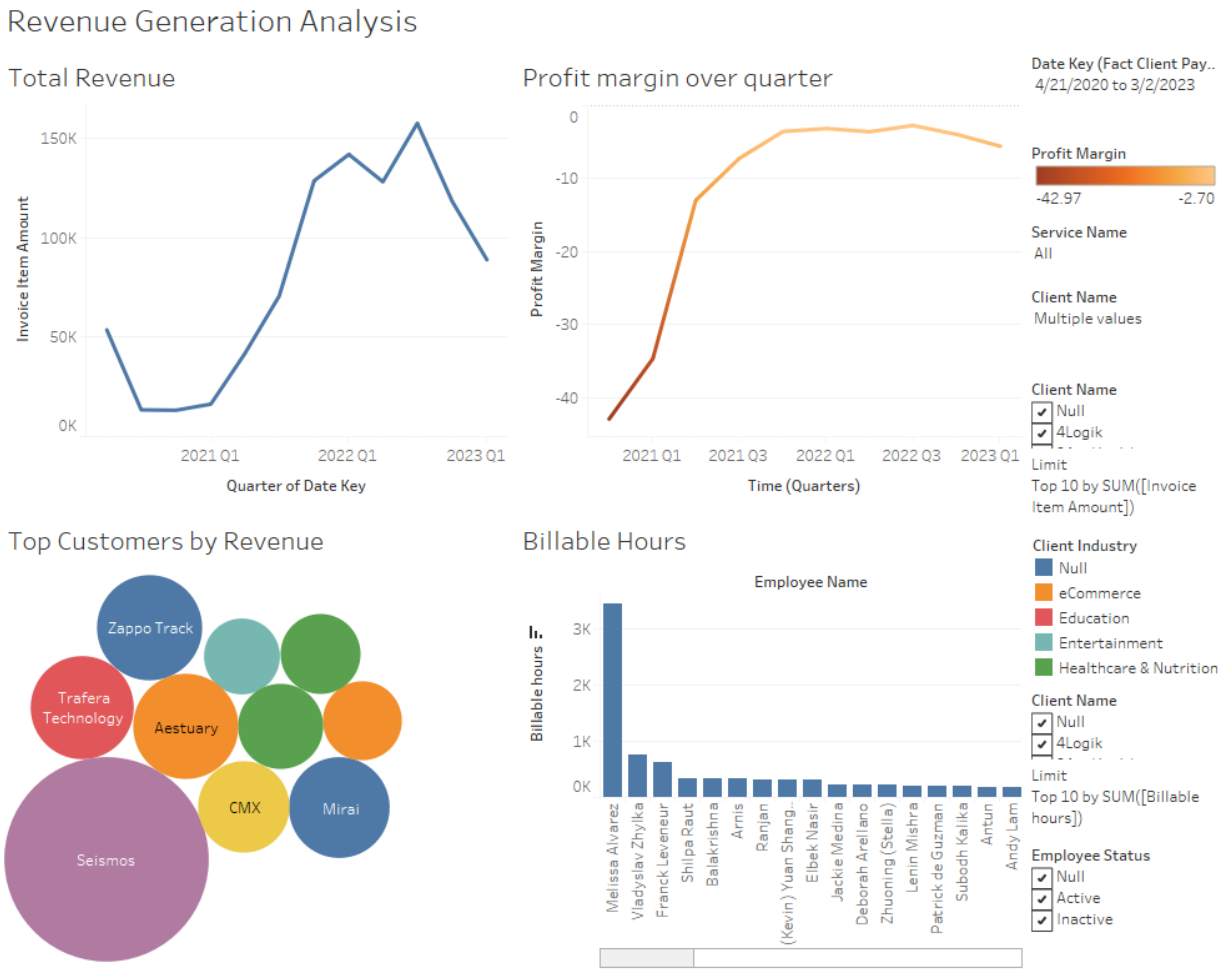
Data Sleek ingests raw data from its source systems, such as Clockify, Monday, Jira, and Quickbooks, into the Data Sleek Data Warehouse (`*DATASLEEK_DW*`). The raw data from the four source systems is brought into the data warehouse by Fivetran and Snowpipe pipelines. It is then transformed, cleaned, and organized using dbt. It is at this stage in the process where we began our querying, analysis, and development of dashboards for the greater purpose of business intelligence and informing decisions.

Our approach to transforming the data from the provided dimensions and fact tables varied by KPI. In certain instances, we created aggregate tables to properly display the data we needed to pull from different sources. In other KPIs, we simply used basic SQL queries to display the necessary data already aggregated in the fact tables. Included within our appendix below are the detailed queries we executed to transform the data into our preferred KPIs. The vast majority of our queries leveraged the fact tables provided by Data Sleek as they contained the relevant information needed for our KPIs.

KPIs and Visualizations

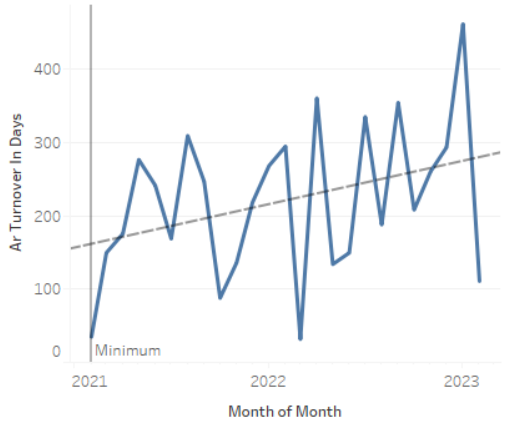
We developed an analysis dashboard using Tableau, which involved selecting essential metrics and KPIs, generating visual representations that offer actionable insights, and identifying significant organizational data that requires ongoing monitoring.

Dashboard Page 1

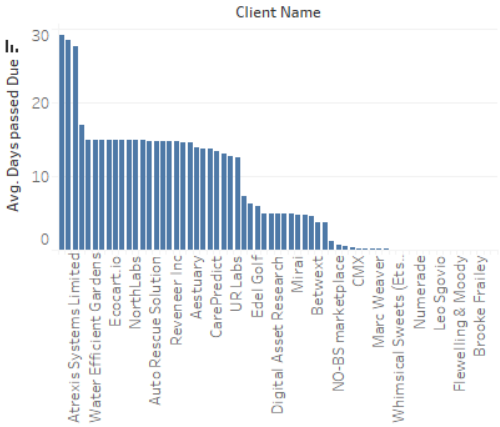


Dashboard Page 2

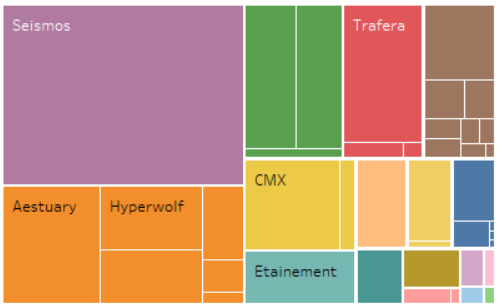
AR Turnover



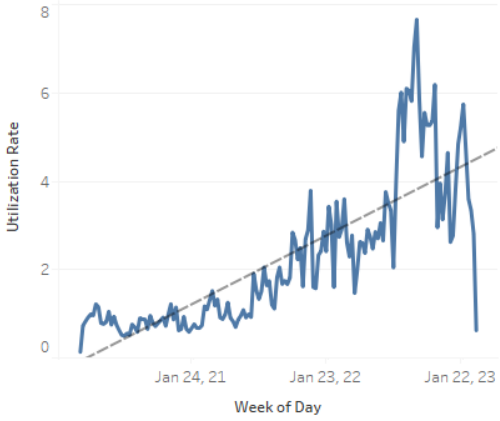
Days past Due

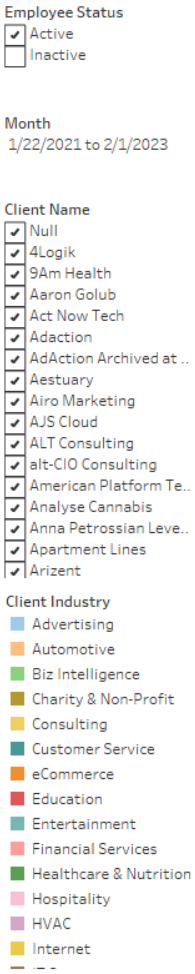


Contract Size per client



Utilization Rate

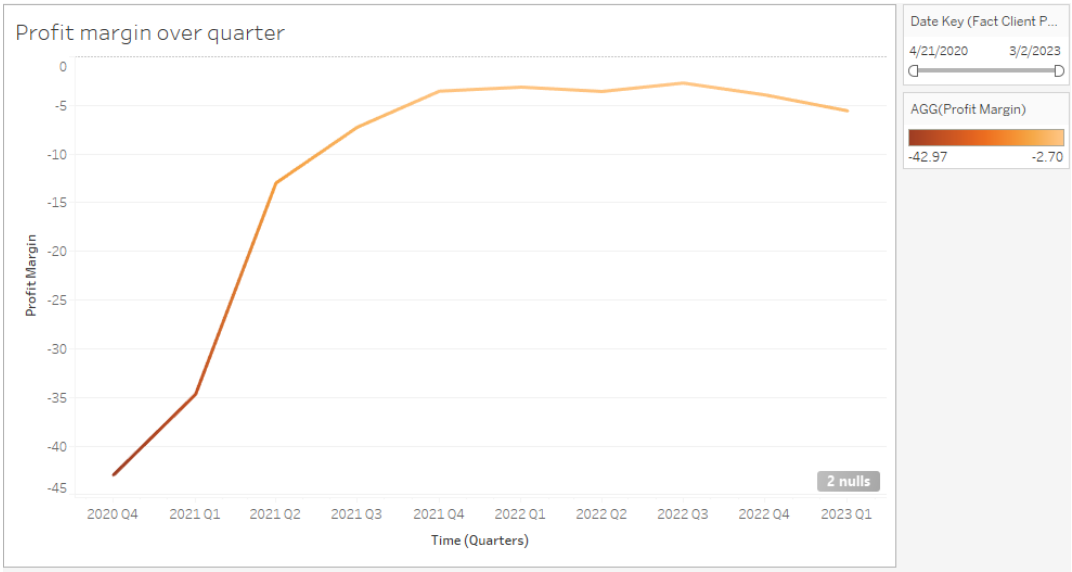




Explanations of Individual Visualizations

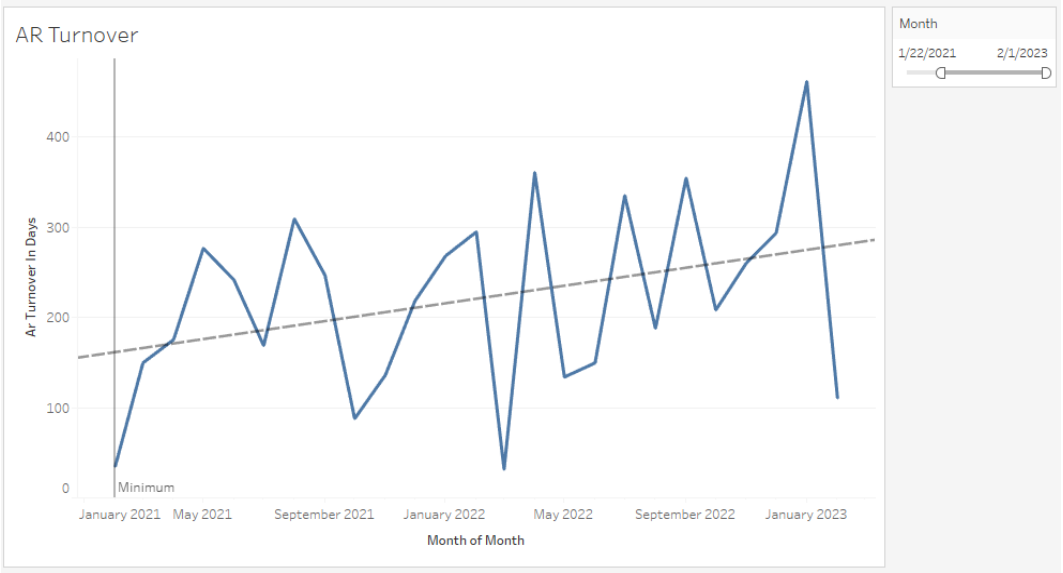
1. Profit Margin by Quarter

The Profit Margin tab calculates the proportion of revenue that remains after deducting all expenses. A higher profit margin signifies improved profitability, and we chose to apply filters to this KPI based on time periods such as quarterly, monthly, yearly, etc.



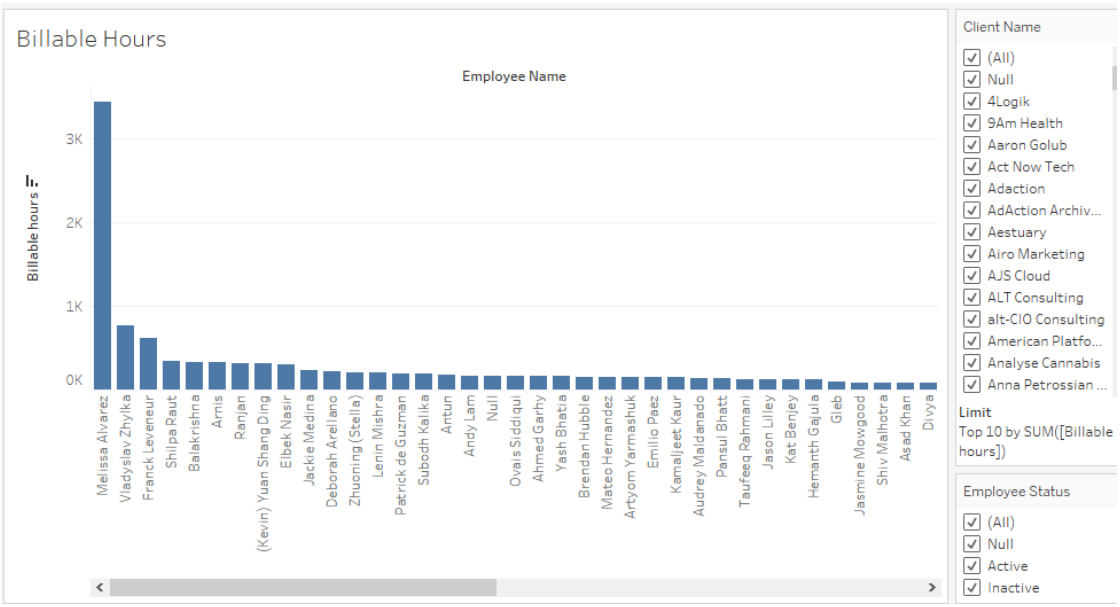
2. Accounts Receivable Turnover

The following Tableau tab enables the user to visualize Data Sleek's Accounts Receivable Turnover (measured in days), which quantifies the rate at which the company receives payments from its customers. A higher accounts receivable turnover indicates a more efficient payment collection process.



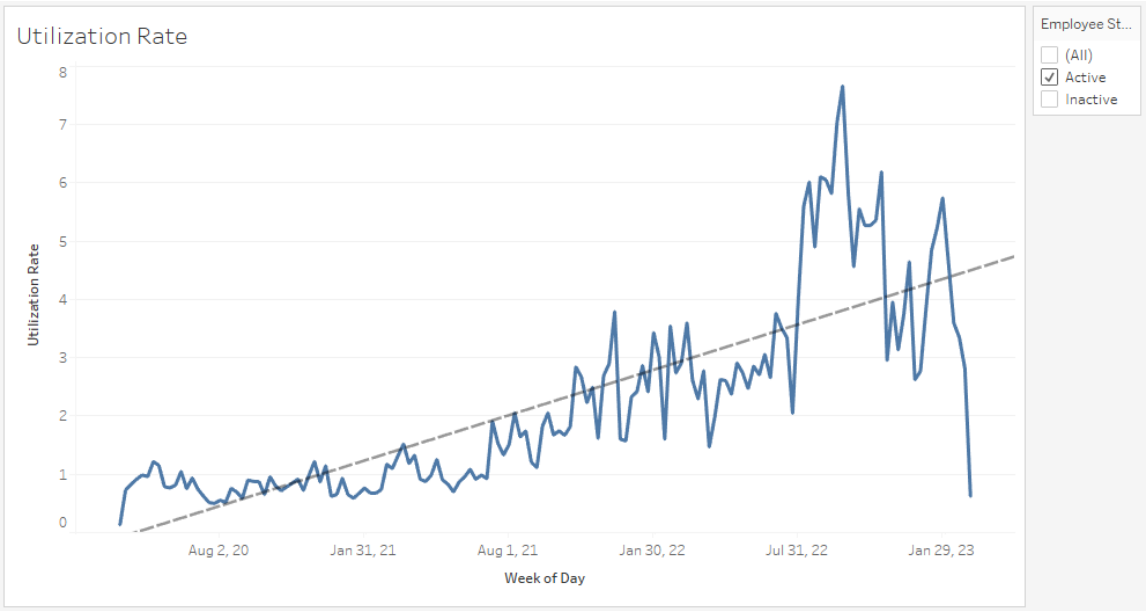
3. Billable Hours

Next, the KPI tab for Billable Hours reports the amount of time that DS consultants have billed to clients. This KPI assists in monitoring billable hours for each consultant and producing reports that indicate the overall billable hours during a given period. The user can filter by employee (active/inactive) or by client.



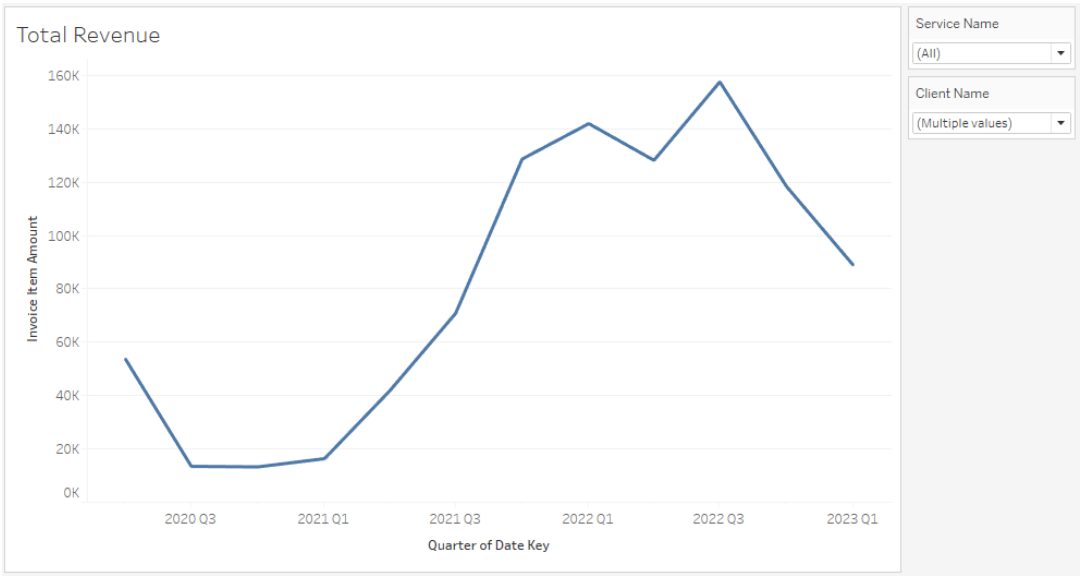
4. Utilization Rate

The following KPI, Utilization Rate, quantifies the proportion of billable hours attributed to a specific client/project within a designated period, by contrasting it against the total available hours. We displayed this in a rolling time series format, and allowed the user to filter by active or inactive employees.



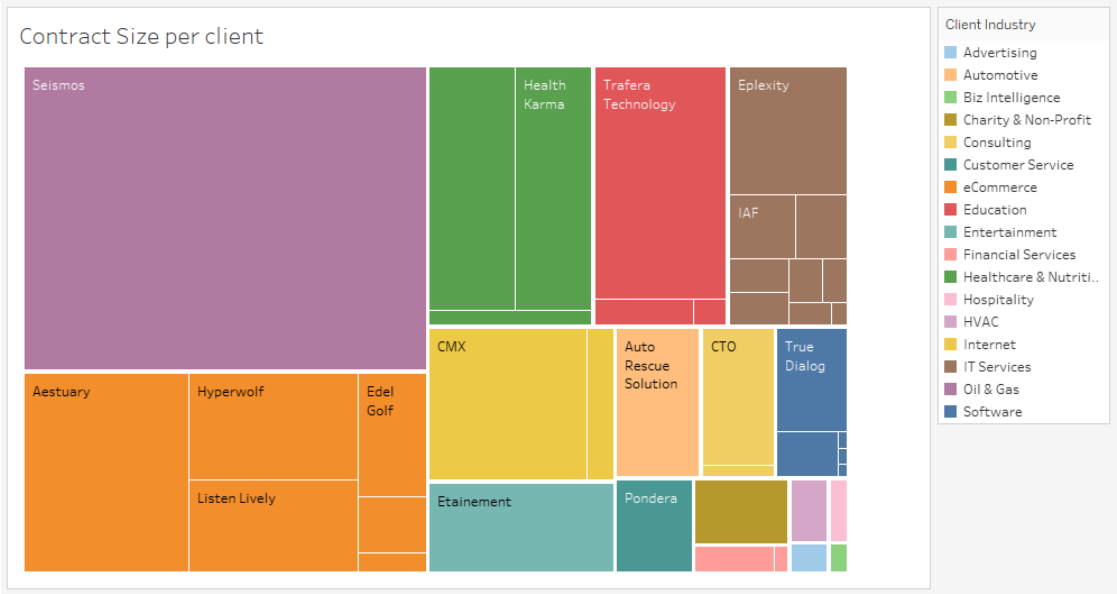
5. Total Revenue

Our Total Revenue KPI evaluates the overall income generated by Data Sleek from various sources, including sales and services. This tab permits the user to view revenue by time and filter by specific Data Sleek service offering or client. Please note that sorting by client is also available on a subsequent KPI tab.



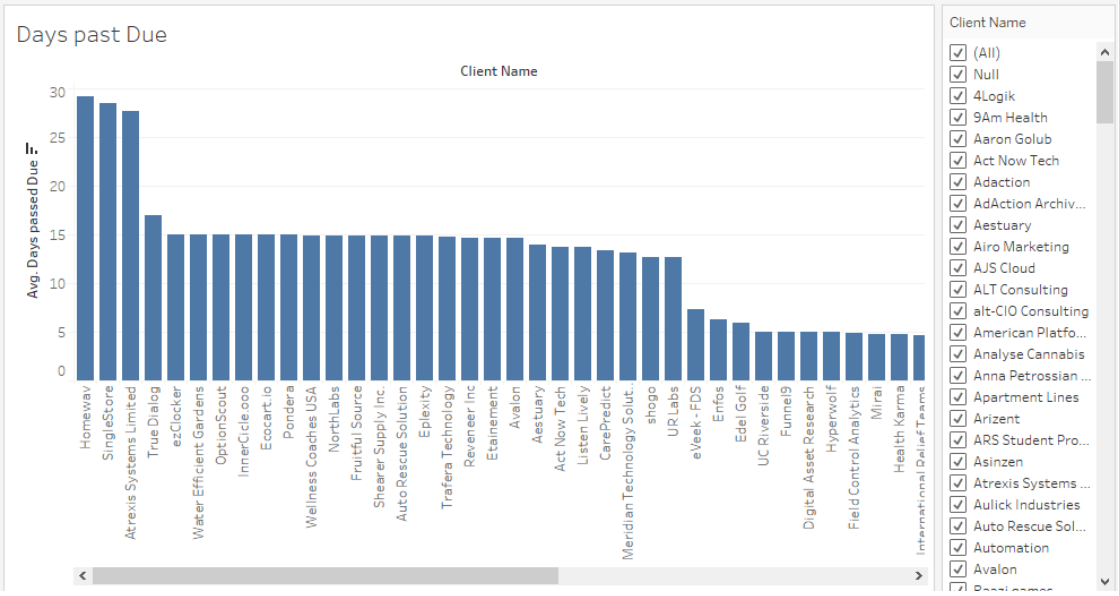
6. Average Contract Size

The first of three KPIs developed by our team is the Average Contract Size per client, which is displayed below. This particular KPI evaluates the average revenue generated from each Data Sleek client. The calculation involves dividing the total revenue by the total number of invoices issued or ongoing contracts, and can filter by industry.



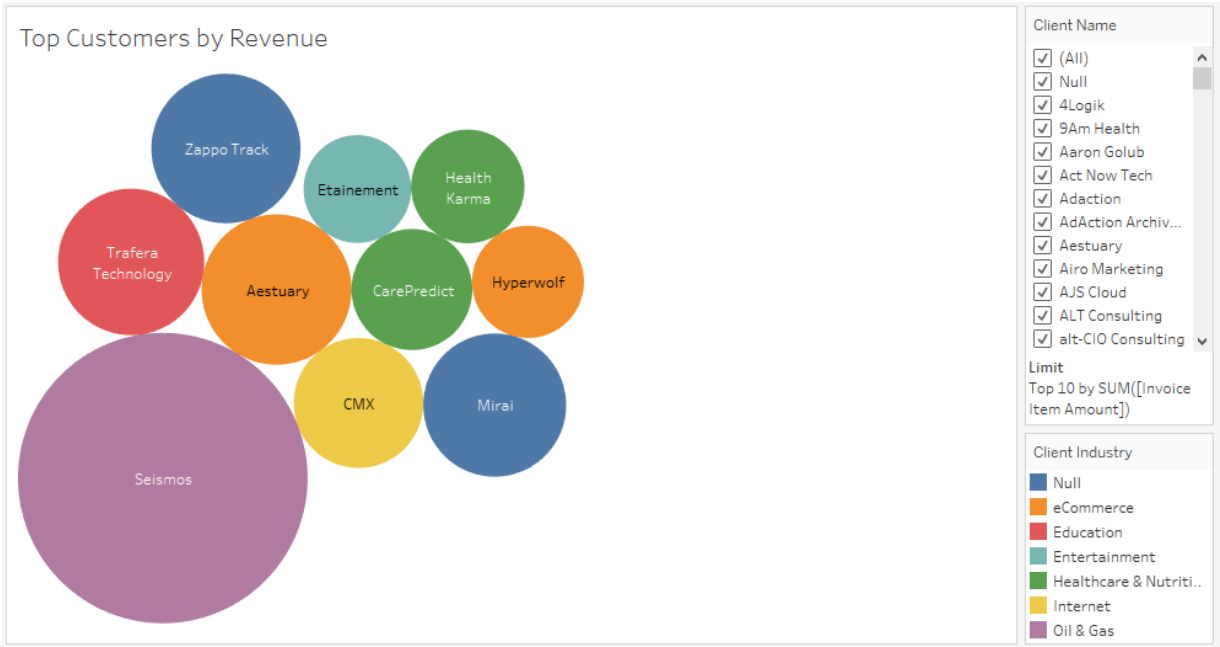
7. Average Days Past Due by Client

The next KPI brainstormed by our team is Days Past Due, or the amount of days clients are overdue on their invoice payments (date of payment receipt - date of payment due date). This KPI helps Data Sleek easily identify any delinquent clients, which allows account managers to be proactive in outreach once a client is flagged.



8. Top Customers By Revenue

Our final KPI of eight is Top Customers by Revenue, which showcases the total amount of revenue generated by the top clients of Data Sleek. The sheet also allows users to filter by client industry.



Insights

Data Sleek's dashboard provides valuable insights into various aspects of the company's operations. In this report, we have broken down our insights into four main categories: revenue and profit trends, top revenue generating clients, billing collections, and employees.

Revenue & Profit Margin:

Total revenue saw a sharp increase during the 2021 Q1 to 2022 Q1 period, and reached its peak during the 2nd half of 2022 before showing a declining trend since then. This is in line with macroeconomic conditions, with the technology sector experiencing a boom during the pandemic, followed by a slowdown in funding and internal expansion.

The profit margin sharply increased during 2021 and then largely plateaued. The decline in profit margin is less steep than total revenue. This suggests that Data Sleek is charging higher prices, but demand is declining. We recommend further investigation into whether this decline is due to macroeconomic trends or an influence of pricing.

Top Clients:

Based on revenue, Seismos is DataSleek's top customer, with a revenue share that is much larger than the next largest customer. This is also reflected in the average contract size of Seismos, which appears to be much larger than our other customers.

Not surprisingly, the top customers by revenue tend to be the same customers with the largest contract sizes. However, we noticed that Zappo Track and Mirai, who bring in some of the highest revenue, do not have a client industry tagged. We recommend that client data, especially for the biggest clients, be properly maintained for optimal client interactions and targeting.

Billing Collections:

Accounts receivable turnover has been on the rise. This indicates that clients are becoming more tardy in making payments. A deeper dive into days past due of each client presents us with a list of our latest payers. Thankfully, the list of offenders is not large, meaning most of Data Sleek's clients do pay on time. Unfortunately, some of our worst offenders include our top revenue generators such as Aestuary and Mirai. We recommend that the collections process be reviewed and possibly improved, to ensure timely payment from all clients.

Employees:
When ranking billable hours by employee, Melissa Alvarez tops the list, with over 3x the billed hours of the second-ranked employee. A further dive into the employee data shows she is the oldest active employee and should thus be awarded the 'Most Loyal Employee' award! We recommend acknowledging and rewarding top-performing employees like Melissa, in order to boost employee morale and retention.

Project Challenges

Throughout the execution of this project, several limitations and challenges were encountered. One of the primary obstacles involved the intricate and vast amount of data required to analyze for the interpretation of key performance indicators (KPIs). This complexity made it difficult to comprehend the tables without access to a data dictionary, thus requiring us to examine each table individually, resulting in the making of several assumptions. Importing data into Tableau also presented issues, frequently resulting in failures or requiring restarts.

Another challenge faced by the team was the selection and implementation of relevant KPIs that aligned with the business objectives and expanded on their definitions. This selection process was somewhat subjective, relying on the team's assumptions about the client's priorities and the granularity of data required for analysis, covering the entire data period from 2019 to 2023. The project also encountered null values in several tables, requiring their exclusion or, in some cases, the removal of entire columns from the dashboard.

To mitigate these challenges, our team made informed assumptions based on available information. When in doubt, we sought clarification from the project coordinators to better inform our decisions and avoid mistakes. The Data Sleek Slack channel was incredibly helpful and a collaborative space across the many teams competing these projects. We also employed a variety of tools and techniques to make sense of the data and select relevant KPIs.

Despite the limitations and challenges faced, our team ultimately overcame them and produced what we believe to be a successful deliverable.

Appendix (SQL KPI Code Snippets)

```
-- Profit Margin KPI SQL Code --

SELECT (1 - ((SELECT sum(vendor_payment_amount)
FROM fact_vendor_payment fv) +
(SELECT sum(employee_payment_amount)
FROM fact_employee_payment fe)) /
(SELECT sum(invoice_item_amount)
FROM fact_invoice_item fi)) *100 as total_profit_margin;

WITH tvp as
(SELECT sum(vendor_payment_amount) as total_vendor_payment, year(date_key) as year
FROM fact_vendor_payment fv
group by year(date_key)),
tep as
(SELECT sum(employee_payment_amount) as total_employee_payment, year(date_key) as year
FROM fact_employee_payment fe
group by year(date_key)),
tr as
(SELECT sum(invoice_item_amount) as total_revenue, year(date_key) as year
FROM fact_invoice_item fi
group by year(date_key))
SELECT tr.year, (1-(total_vendor_payment+total_employee_payment)/total_revenue)*100 as profit_margin_by_year
FROM tr
JOIN tep on tr.year = tep.year
JOIN tvp on tr.year = tvp.year;
```

```
-- Accounts Receivable Turnover KPI SQL Code --

SELECT SALES2.MONTH, NET_SALES, ROLLING_AR AS AR, NET_SALES/ROLLING_AR AS AR_TURNOVER,
365/(NET_SALES/ROLLING_AR) AS AR_TURNOVER_IN_DAYS
FROM
   (SELECT DATE_TRUNC('month', date_key) AS MONTH, sum(item_amount) AS NET_SALES
   FROM
      (
      SELECT invoice_item_key, c.client_name AS "Client Name", s.service_name AS "Service Type", invoice_key, date_key,
sum(invoice_item_payment_price),
      sum(invoice_item_quantity), sum(invoice_item_amount) as item_amount
      FROM FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.fact_invoice_item f
      JOIN FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.dim_service s ON s.service_key = f.service_key
      JOIN FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.dim_client c on c.client_key = f.client_key
      GROUP BY invoice_item_key, c.client_name, s.service_name, invoice_key, date_key
      ) AS sales
   GROUP BY 1
   ORDER BY 1) AS SALES2
JOIN
   (SELECT
      invoice.month, total_invoice_amount,payment_in_month, total_invoice_amount-payment_in_month AS AR_CHANGE,
      SUM(AR_CHANGE) OVER (ORDER BY INVOICE.MONTH) AS ROLLING_AR
   FROM
         (SELECT DATE_TRUNC('month', di.invoice_due_date) as month ,sum(fi.invoice_item_amount) as total_invoice_amount
         FROM
         FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.fact_invoice_item fi
         JOIN FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.dim_invoice di ON fi.invoice_key = di.invoice_key
```

```
        GROUP BY 1) AS INVOICE
    LEFT JOIN
        (SELECT DATE_TRUNC('month', DATE_KEY) AS MONTH, SUM(client_payment_amount) AS payment_in_month
        FROM
        FWLOYYD_PZA82601_DATASLEEK_DW_UCLA.DS_DWH_UCLA.fact_client_payment
        GROUP BY 1) AS PAYMENT
    ON invoice.month = PAYMENT.MONTH
    WHERE INVOICE.MONTH IS NOT NULL
    ORDER BY 1) AS AR
ON SALES2.MONTH = AR.MONTH
WHERE SALES2.MONTH <= '2023-02-01'
ORDER BY 1;
```

```
-- Billable Hours KPI SQL Code --

-- Total billable hours
SELECT sum(invoice_item_quantity) as billable_hours
FROM fact_invoice_item fi;
-- Billable hours by clients
SELECT client_name,
sum(invoice_item_quantity) as billable_hours
FROM fact_invoice_item fi
JOIN dim_client dc ON fi.client_key = dc.client_key
group by 1;
```

```
-- Utilization Rate KPI SQL Code --

SELECT
(SELECT sum(invoice_item_quantity) as billable_hours
FROM fact_invoice_item fi)
/
(SELECT sum(employee_capacity_hours)
FROM dim_employee)
*100 as total_utilization_rate;
```

```
-- Total Revenue KPI SQL Code --

SELECT invoice_item_key, c.client_name AS "Client Name", s.service_name AS "Service Type", invoice_key, date_key,
sum(invoice_item_payment_price), sum(invoice_item_quantity), sum(invoice_item_amount)
FROM fact_invoice_item f
JOIN dim_service s ON s.service_key = f.service_key
JOIN dim_client c on c.client_key = f.client_key
GROUP BY invoice_item_key, c.client_name, s.service_name, invoice_key, date_key;
```

```
-- Average Invoice Amount KPI SQL Code --

select AVG(invoice_item_amount), date_key
FROM fact_invoice_item
GROUP BY date_key;
```

```sql
-- Days Past Due KPI SQL Code --

SELECT
client_name,
fi.invoice_key,
fi.date_key as invoice_date,
di.invoice_due_date,
COALESCE(fc.date_key, GETDATE()) AS client_payment_date,
DATEDIFF(DAY, di.invoice_due_date, COALESCE(fc.date_key, GETDATE())) AS days_past_due
FROM fact_invoice_item fi
LEFT JOIN fact_client_payment fc ON fi.invoice_key = fc.invoice_key
JOIN dim_client dc ON fi.client_key = dc.client_key
JOIN dim_invoice di ON fi.invoice_key = di.invoice_key
WHERE days_past_due > 0
group by 1,2,3,4,5;
```

```sql
-- Top Revenue Customers KPI SQL Code --


select client_name, sum(invoice_item_amount)
from fact_invoice_item fi
join dim_client dc ON fi.client_key = dc.client_key
group by client_name
order by 2 DESC limit 10;
```