DATA MANAGEMENT - FINAL PROJECT 2024

PROJECT 3

GROUP 18

Mrigangka Medhi (606309858)
Yuqi Zhao (606310300)
Alexandar Beltchev (606318438)
Jingxin (Jessica) Tang (706310309)
Oscar Trumpy (706310371)

Table of Contents

Executive Summary

In this project, we embarked on an exploration of agile project management through the lens of Jira Software by Atlassian, with the goal of enhancing the project management process through data-driven analysis. We aimed to create a dynamic analytical framework in Snowflake that reflects the current status of projects and predicts future trends, utilizing the Data Sleek data warehouse.

First, our approach involved detailed data analysis, starting with SQL queries to dissect key aspects of project management, such as issue tracking, user participation, and project progression. To make this more clear, we developed a comprehensive data dictionary and constructed an Entity Relation Diagram (ERD) to ensure a more in-depth understanding and accurate representation of the Jira dataset.

Through dimensional modeling, we designed dimension and fact tables to capture the critical elements of project management, and this structure allowed us to obtain key performance indicators (KPIs) that measure various aspects of project and user activities, providing a quantitative foundation for our analysis.

After Dissecting multiple parts of the dataset, here, we successfully identified:

- The number of issues created, resolved, and closed per project highlighted workflow efficiencies and identified potential areas for improvement.
- User activity KPIs that help us understand how much each person contributed, showing us who did really well and where we might need to offer more training or help.
- A comprehensive view of project health and operational effectiveness by analyzing project progress and status through KPIs related to issue statuses, priorities, and due dates.

To convey our insights, we utilized Tableau for data visualization, creating an interactive display that made our findings more transparent and more understandable to a broader audience. Despite encountering challenges like data transformation and the intricacies of data interpretation, we successfully uncovered important insights into the agile project management process. Our project highlighted the importance of a data-driven approach in agile project management, providing actionable insights into project performance and guiding strategic decision-making for continuous improvement in project management efficiency.

Project Statement

Agile project management is rapidly becoming an essential component of successful software development and delivery. Jira Software is a project management tool designed by Atlassian. It provides an efficient way for organizations to manage the project development process, as well as facilitating planning, tracking, and releasing. Instead of traditional methods that often lag in periodic data evaluation and action, this project focuses on the dynamic flow of project-related activities. By integrating the data source from the Data Sleek data warehouse, we aim to build an analytical framework that reflects the project status and future predictions.

In this project, we used Data Sleek's data resources to write SQL queries for dissecting aspects of project management such as issue tracking, user participation, and project progression, therefore improving the efficiency and productivity of Jira Software. The dimension and fact tables that we created allow us to translate the analytical process into key performance indicators (KPIs) to measure the project management process. In the end, we summarized our analysis through Tableau visualization, where we displayed insights to present the project management performance based on Jira's dataset.

Data Dictionary

In this project, we have developed a detailed data dictionary to outline the structure of the JIRA dataset. The data dictionary catalogs each table and its respective columns, detailing data types, nullability, and ordinal positions to ensure data integrity and consistency across the database. The below table presents all attributes from all 31 tables in the Jira dataset.

TABLE_NAME	COLUMN_NAME	DATA_TYPE	ORDINAL_POSITION	IS_NULLABLE
BOARD	ID	NUMBER	1	NO
BOARD	NAME	TEXT	2	YES
BOARD	TYPE	TEXT	3	YES
BOARD	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
BOARD	_FIVETRAN_DELETED	BOOLEAN	5	YES
COMMENT	ID	NUMBER	1	NO
COMMENT	ISSUE_ID	NUMBER	2	YES
COMMENT	AUTHOR_ID	TEXT	3	YES
COMMENT	UPDATE_AUTHOR_ID	TEXT	4	YES
COMMENT	BODY	TEXT	5	YES

COMMENT	CREATED	TIMESTAMP_TZ	6	YES
COMMENT	UPDATED	TIMESTAMP_TZ	7	YES
COMMENT	IS_PUBLIC	BOOLEAN	8	YES
COMMENT	_FIVETRAN_SYNCED	TIMESTAMP_TZ	9	YES
EPIC	ID	NUMBER	1	NO
EPIC	KEY	TEXT	2	YES
EPIC	NAME	TEXT	3	YES
EPIC	SUMMARY	TEXT	4	YES
EPIC	DONE	BOOLEAN	5	YES
EPIC	_FIVETRAN_SYNCED	TIMESTAMP_TZ	6	YES
FIELD	ID	TEXT	1	NO
FIELD	NAME	TEXT	2	YES
FIELD	IS_CUSTOM	BOOLEAN	3	YES
FIELD	IS_ARRAY	BOOLEAN	4	YES
FIELD	_FIVETRAN_DELETED	BOOLEAN	5	YES
FIELD	_FIVETRAN_SYNCED	TIMESTAMP_TZ	6	YES
FIELD_OPTION	ID	NUMBER	1	NO
FIELD_OPTION	PARENT_ID	NUMBER	2	YES
FIELD_OPTION	NAME	TEXT	3	YES
FIELD_OPTION	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
FIVETRAN_AUDIT	ID	TEXT	1	NO
FIVETRAN_AUDIT	MESSAGE	TEXT	2	YES
FIVETRAN_AUDIT	UPDATE_STARTED	TIMESTAMP_TZ	3	YES
FIVETRAN_AUDIT	UPDATE_ID	TEXT	4	YES
FIVETRAN_AUDIT	SCHEMA	TEXT	5	YES
FIVETRAN_AUDIT	TABLE	TEXT	6	YES
FIVETRAN_AUDIT	START	TIMESTAMP_TZ	7	YES
FIVETRAN_AUDIT	DONE	TIMESTAMP_TZ	8	YES
FIVETRAN_AUDIT	ROWS_UPDATED_OR_INSERTED	NUMBER	9	YES
FIVETRAN_AUDIT	STATUS	TEXT	10	YES
FIVETRAN_AUDIT	PROGRESS	TIMESTAMP_TZ	11	YES
FIVETRAN_AUDIT	_FIVETRAN_SYNCED	TIMESTAMP_TZ	12	YES
ISSUE	ID	NUMBER	1	NO

ISSUE	KEY	TEXT	2	YES
ISSUE	PARENT_ID	NUMBER	3	YES
ISSUE	STATUS_CATEGORY_CHANGED	TIMESTAMP_TZ	4	YES
ISSUE	ISSUE_TYPE	NUMBER	5	YES
ISSUE	TIME_SPENT	FLOAT	6	YES
ISSUE	PROJECT	NUMBER	7	YES
ISSUE	_TIME_SPENT	FLOAT	8	YES
ISSUE	RESOLUTION	NUMBER	9	YES
ISSUE	RESOLVED	TIMESTAMP_TZ	10	YES
ISSUE	WORK_RATIO	FLOAT	11	YES
ISSUE	LAST_VIEWED	TIMESTAMP_TZ	12	YES
ISSUE	CREATED	TIMESTAMP_TZ	13	YES
ISSUE	PRIORITY	NUMBER	14	YES
ISSUE	REMAINING_ESTIMATE	FLOAT	15	YES
ISSUE	_ORIGINAL_ESTIMATE	FLOAT	16	YES
ISSUE	ASSIGNEE	TEXT	17	YES
ISSUE	UPDATED	TIMESTAMP_TZ	18	YES
ISSUE	STATUS	NUMBER	19	YES
ISSUE	ORIGINAL_ESTIMATE	FLOAT	20	YES
ISSUE	DESCRIPTION	TEXT	21	YES
ISSUE	SECURITY_LEVEL	NUMBER	22	YES
ISSUE	_REMAINING_ESTIMATE	FLOAT	23	YES
ISSUE	SUMMARY	TEXT	24	YES
ISSUE	CREATOR	TEXT	25	YES
ISSUE	REPORTER	TEXT	26	YES
ISSUE	ENVIRONMENT	TEXT	27	YES
ISSUE	DUE_DATE	DATE	28	YES
ISSUE	_FIVETRAN_DELETED	BOOLEAN	29	YES
ISSUE	_FIVETRAN_SYNCED	TIMESTAMP_TZ	30	YES
ISSUE	PARENT	NUMBER	31	YES
ISSUE_BOARD	BOARD_ID	NUMBER	1	NO
ISSUE_BOARD	ISSUE_ID	NUMBER	2	NO
ISSUE_BOARD	_FIVETRAN_SYNCED	TIMESTAMP_TZ	3	YES

ISSUE_BOARD	_FIVETRAN_DELETED	BOOLEAN	4	YES
ISSUE_FIELD_HISTORY	FIELD_ID	TEXT	1	NO
ISSUE_FIELD_HISTORY	ISSUE_ID	NUMBER	2	NO
ISSUE_FIELD_HISTORY	TIME	TIMESTAMP_TZ	3	NO
ISSUE_FIELD_HISTORY	VALUE	TEXT	4	YES
ISSUE_FIELD_HISTORY	IS_ACTIVE	BOOLEAN	5	YES
ISSUE_FIELD_HISTORY	AUTHOR_ID	TEXT	6	YES
ISSUE_FIELD_HISTORY	_FIVETRAN_SYNCED	TIMESTAMP_TZ	7	YES
ISSUE_LINK	ISSUE_ID	NUMBER	1	NO
ISSUE_LINK	RELATED_ISSUE_ID	NUMBER	2	NO
ISSUE_LINK	RELATIONSHIP	TEXT	3	NO
ISSUE_LINK	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
ISSUE_MULTISELECT_HISTORY	FIELD_ID	TEXT	1	NO
ISSUE_MULTISELECT_HISTORY	ISSUE_ID	NUMBER	2	NO
ISSUE_MULTISELECT_HISTORY	TIME	TIMESTAMP_TZ	3	NO
ISSUE_MULTISELECT_HISTORY	_FIVETRAN_ID	TEXT	4	NO
ISSUE_MULTISELECT_HISTORY	VALUE	TEXT	5	YES
ISSUE_MULTISELECT_HISTORY	IS_ACTIVE	BOOLEAN	6	YES
ISSUE_MULTISELECT_HISTORY	AUTHOR_ID	TEXT	7	YES
ISSUE_MULTISELECT_HISTORY	_FIVETRAN_SYNCED	TIMESTAMP_TZ	8	YES
ISSUE_TYPE	ID	NUMBER	1	NO
ISSUE_TYPE	NAME	TEXT	2	YES
ISSUE_TYPE	DESCRIPTION	TEXT	3	YES

ISSUE_TYPE	SUBTASK	BOOLEAN	4	YES
ISSUE_TYPE	_FIVETRAN_SYNCED	TIMESTAMP_TZ	5	YES
ISSUE_WATCHER	ISSUE_ID	NUMBER	1	NO
ISSUE_WATCHER	USER_ID	TEXT	2	NO
ISSUE_WATCHER	_FIVETRAN_SYNCED	TIMESTAMP_TZ	3	YES
PERMISSION	ID	TEXT	1	NO
PERMISSION	NAME	TEXT	2	YES
PERMISSION	TYPE	TEXT	3	YES
PERMISSION	DESCRIPTION	TEXT	4	YES
PERMISSION	_FIVETRAN_DELETED	BOOLEAN	5	YES
PERMISSION	_FIVETRAN_SYNCED	TIMESTAMP_TZ	6	YES
PERMISSION_HOLDER	_FIVETRAN_ID	TEXT	1	NO
PERMISSION_HOLDER	PERMISSION_SCHEME_ID	NUMBER	2	YES
PERMISSION_HOLDER	PERMISSION_ID	TEXT	3	YES
PERMISSION_HOLDER	USER_ID	TEXT	4	YES
PERMISSION_HOLDER	GROUP_NAME	TEXT	5	YES
PERMISSION_HOLDER	PROJECT_ROLE_ID	NUMBER	6	YES
PERMISSION_HOLDER	USER_CUSTOM_FIELD	TEXT	7	YES
PERMISSION_HOLDER	GROUP_CUSTOM_FIELD	TEXT	8	YES
PERMISSION_HOLDER	TYPE	TEXT	9	YES
PERMISSION_HOLDER	_FIVETRAN_SYNCED	TIMESTAMP_TZ	10	YES
PERMISSION_HOLDER	_FIVETRAN_DELETED	BOOLEAN	11	YES
PERMISSION_SCHEME	ID	NUMBER	1	NO
PERMISSION_SCHEME	NAME	TEXT	2	YES

PERMISSION_SCHEME	DESCRIPTION	TEXT	3	YES
PERMISSION_SCHEME	_FIVETRAN_DELETED	BOOLEAN	4	YES
PERMISSION_SCHEME	_FIVETRAN_SYNCED	TIMESTAMP_TZ	5	YES
PRIORITY	ID	NUMBER	1	NO
PRIORITY	NAME	TEXT	2	YES
PRIORITY	DESCRIPTION	TEXT	3	YES
PRIORITY	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
PROJECT	ID	NUMBER	1	NO
PROJECT	KEY	TEXT	2	YES
PROJECT	NAME	TEXT	3	YES
PROJECT	PROJECT_TYPE_KEY	TEXT	4	YES
PROJECT	DESCRIPTION	TEXT	5	YES
PROJECT	PROJECT_CATEGORY_ID	NUMBER	6	YES
PROJECT	LEAD_ID	TEXT	7	YES
PROJECT	PERMISSION_SCHEME_ID	NUMBER	8	YES
PROJECT	_FIVETRAN_DELETED	BOOLEAN	9	YES
PROJECT	_FIVETRAN_SYNCED	TIMESTAMP_TZ	10	YES
PROJECT_BOARD	BOARD_ID	NUMBER	1	NO
PROJECT_BOARD	PROJECT_ID	NUMBER	2	NO
PROJECT_BOARD	_FIVETRAN_SYNCED	TIMESTAMP_TZ	3	YES
PROJECT_BOARD	_FIVETRAN_DELETED	BOOLEAN	4	YES
PROJECT_ROLE	ID	NUMBER	1	NO
PROJECT_ROLE	NAME	TEXT	2	YES
PROJECT_ROLE	DESCRIPTION	TEXT	3	YES
PROJECT_ROLE	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
PROJECT_ROLE_ACTOR	ID	NUMBER	1	NO
PROJECT_ROLE_ACTOR	PROJECT_ID	NUMBER	2	YES
PROJECT_ROLE_ACTOR	PROJECT_ROLE_ID	NUMBER	3	YES
PROJECT_ROLE_ACTOR	USER_ID	TEXT	4	YES

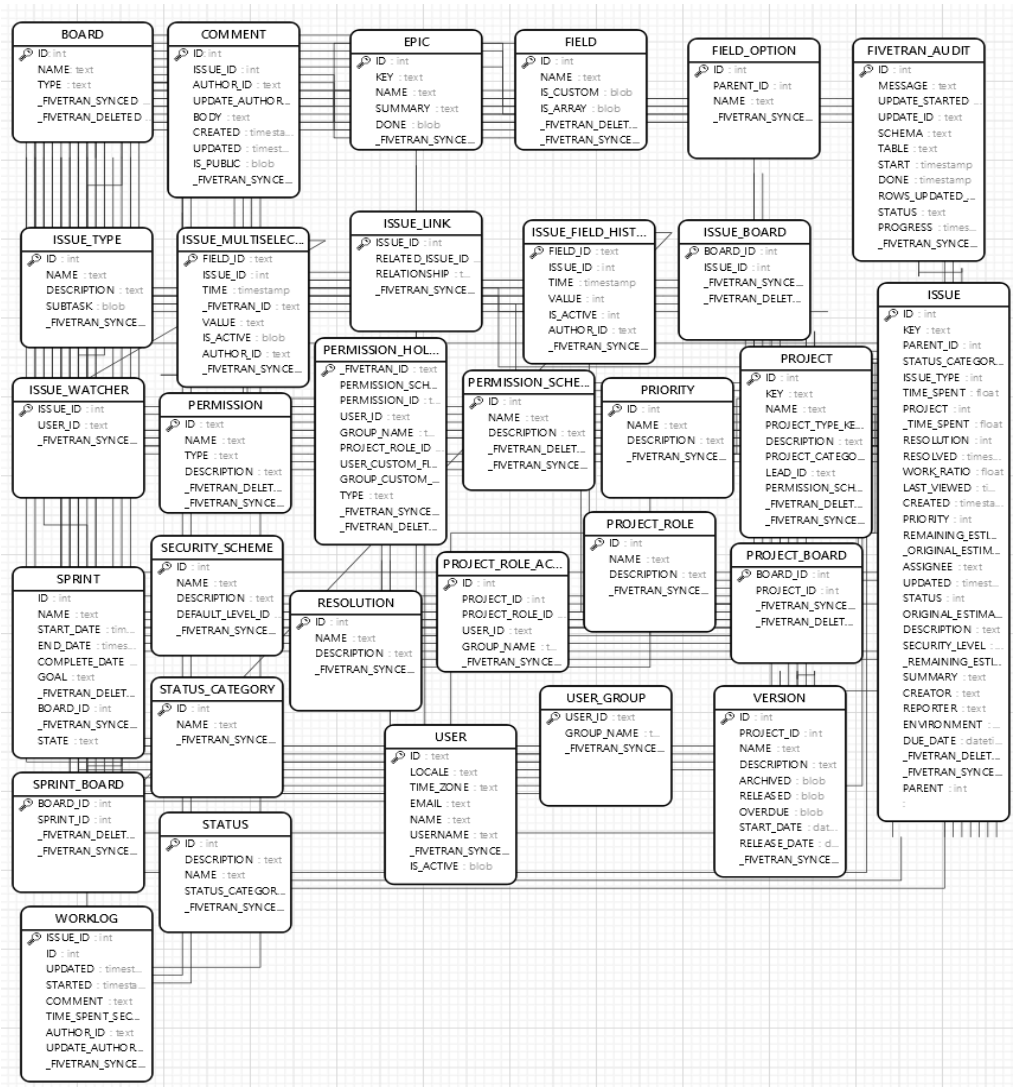
PROJECT_ROLE_ACTOR	GROUP_NAME	TEXT	5	YES
PROJECT_ROLE_ACTOR	_FIVETRAN_SYNCED	TIMESTAMP_TZ	6	YES
RESOLUTION	ID	NUMBER	1	NO
RESOLUTION	NAME	TEXT	2	YES
RESOLUTION	DESCRIPTION	TEXT	3	YES
RESOLUTION	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
SECURITY_SCHEME	ID	NUMBER	1	NO
SECURITY_SCHEME	NAME	TEXT	2	YES
SECURITY_SCHEME	DESCRIPTION	TEXT	3	YES
SECURITY_SCHEME	DEFAULT_LEVEL_ID	NUMBER	4	YES
SECURITY_SCHEME	_FIVETRAN_SYNCED	TIMESTAMP_TZ	5	YES
SPRINT	ID	NUMBER	1	NO
SPRINT	NAME	TEXT	2	YES
SPRINT	START_DATE	TIMESTAMP_TZ	3	YES
SPRINT	END_DATE	TIMESTAMP_TZ	4	YES
SPRINT	COMPLETE_DATE	TIMESTAMP_TZ	5	YES
SPRINT	GOAL	TEXT	6	YES
SPRINT	_FIVETRAN_DELETED	BOOLEAN	7	YES
SPRINT	BOARD_ID	NUMBER	8	YES
SPRINT	_FIVETRAN_SYNCED	TIMESTAMP_TZ	9	YES
SPRINT	STATE	TEXT	10	YES
SPRINT_BOARD	BOARD_ID	NUMBER	1	NO
SPRINT_BOARD	SPRINT_ID	NUMBER	2	NO
SPRINT_BOARD	_FIVETRAN_DELETED	BOOLEAN	3	YES
SPRINT_BOARD	_FIVETRAN_SYNCED	TIMESTAMP_TZ	4	YES
STATUS	ID	NUMBER	1	NO
STATUS	DESCRIPTION	TEXT	2	YES
STATUS	NAME	TEXT	3	YES
STATUS	STATUS_CATEGORY_ID	NUMBER	4	YES

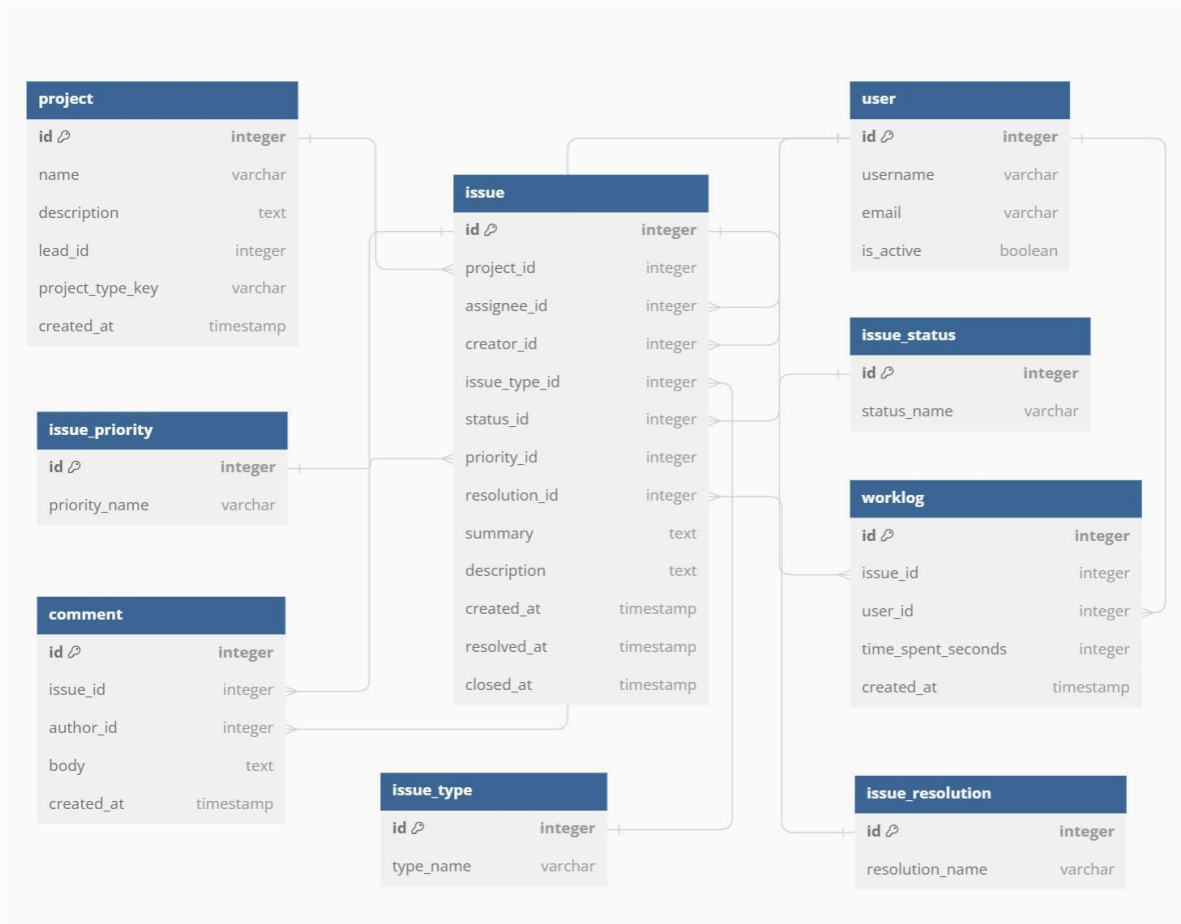
STATUS	_FIVETRAN_SYNCED	TIMESTAMP_TZ	5	YES
STATUS_CATEGORY	ID	NUMBER	1	NO
STATUS_CATEGORY	NAME	TEXT	2	YES
STATUS_CATEGORY	_FIVETRAN_SYNCED	TIMESTAMP_TZ	3	YES
USER	ID	TEXT	1	NO
USER	LOCALE	TEXT	2	YES
USER	TIME_ZONE	TEXT	3	YES
USER	EMAIL	TEXT	4	YES
USER	NAME	TEXT	5	YES
USER	USERNAME	TEXT	6	YES
USER	_FIVETRAN_SYNCED	TIMESTAMP_TZ	7	YES
USER	IS_ACTIVE	BOOLEAN	8	YES
USER_GROUP	USER_ID	TEXT	1	NO
USER_GROUP	GROUP_NAME	TEXT	2	NO
USER_GROUP	_FIVETRAN_SYNCED	TIMESTAMP_TZ	3	YES
VERSION	ID	NUMBER	1	NO
VERSION	PROJECT_ID	NUMBER	2	YES
VERSION	NAME	TEXT	3	YES
VERSION	DESCRIPTION	TEXT	4	YES
VERSION	ARCHIVED	BOOLEAN	5	YES
VERSION	RELEASED	BOOLEAN	6	YES
VERSION	OVERDUE	BOOLEAN	7	YES
VERSION	START_DATE	DATE	8	YES
VERSION	RELEASE_DATE	DATE	9	YES
VERSION	_FIVETRAN_SYNCED	TIMESTAMP_TZ	10	YES
WORKLOG	ISSUE_ID	NUMBER	1	NO
WORKLOG	ID	NUMBER	2	NO
WORKLOG	UPDATED	TIMESTAMP_TZ	3	YES
WORKLOG	STARTED	TIMESTAMP_TZ	4	YES
WORKLOG	COMMENT	TEXT	5	YES
WORKLOG	TIME_SPENT_SECONDS	NUMBER	6	YES
WORKLOG	AUTHOR_ID	TEXT	7	YES

WORKLOG	UPDATE_AUTHOR_ID	TEXT	8	YES
WORKLOG	_FIVETRAN_SYNCED	TIMESTAMP_TZ	9	YES

Based on our data dictionary and all attributes presented, we extracted the critical tables for our analysis in order to construct a comprehensive entity relation diagram (ERD), enabling us to present dimensional modeling with the creation of new dimensional and fact tables. All significant tables we used are designed to make the data easily relatable to the business processes, with considerations for future enhancements. Therefore, this data dictionary serves not only as an understanding of the JIRA database but also as a guideline for the utilization and enhancement of our analysis.

Entity Relation Diagram (ERD)





Note: The second ERD is included for more apparent reference.

Database structure overview

The database represented within the ERD diagrams is an organized collection of tables that are necessary to an application's backend, likely one related to extended following and administration. This database incorporates an arrangement of interrelated tables such as "BOARD", "COMMENT", "EPIC", "FIELD", and "FIELD_OPTION", among others. The connections between these tables, characterized by essential and outside keys, frame a complex however coherent pattern outlined for effective information organization and recovery.

Tables and attributes

The “BOARD” table is an imperative portion of the database. It holds distinctive parts of a venture or a workspace. The code “ID” appears which board it is. The “NAME” property is the board's title, whereas the “TYPE” property puts the board into categories. Metadata traits “FIVETRAN_SYNCED” and “FIVETRAN_DELETED” keep track of whether information has been synchronized and coherently erased. The “COMMENT” table is made to keep track of comments made by users about other things within the database. Each comment has its claim

special number called an “ID” and incorporates words within the “BODY” segment. The ERD is associated with the “USER” and “BOARD” tables. This interfaces a comment to the individual who composed it and the particular board it is for.

In ERD, an “EPIC” stands for an enormous piece of work. The “EPIC” table appears this thought. Each epic features an uncommon “ID” and details like “NAME,” “SUMMARY,” and “OWNER_ID”. The “OWNER_ID” may be a number that interfaces the epic to the client who is dependable for it. This number could be from the “USER” table. The “FIELD” table is utilized to organize things that can be changed in a board or epic. It features a primary ID and areas like Title and Sort. This permits diverse information areas that can be changed to fit distinctive projects. The ERD depicts how distinctive things are associated, like how areas are related to sheets and other things utilizing remote keys.

The “FIELD_OPTION” table makes a difference with the choices for the areas within the “FIELD” table. The “ID” is like an extraordinary key, and the FIELD_ID interfaces to the FIELD table, making them related. The “VALUE” quality contains the information that can be chosen for a field. The “ISSUE” table is exceptionally imperative within the database. It likely holds data approximately diverse assignments, issues, or demands. Each line in this table has its own special number called an “ID” which conjointly encompasses a “KEY” to assist in discovering the issue within the application. The table includes a “SUMMARY” to deliver a brief portrayal, a “STATUS_ID” that interfaces to the “STATUS” table to show the current state of the issue, and other points of interest around need, sort, and associations to clients and other things.

The “SPRINT” table makes a difference in keeping track of dexterous sprints. Sprints are brief periods of time when a group works to wrap up a certain sum of work. Characteristics incorporate “ID”, “NAME”, “START_DATE”, and “END_DATE”, which offer assistance in characterizing the time and scope of each sprint. This table likely works beside the “ISSUE” table to interface issues with certain sprints. This “USER” table has points of interest for almost all the individuals using the framework or the individuals of the group. It contains a distinct “ID” with individual data like “NAME” and “EMAIL”. Numerous other tables might utilize this as an outside key to determine which client is related to a particular thing, such as an issue, comment, or responsibility.

The “PROJECT” table contains subtle elements approximately each venture within the application. Each extend has its claim extraordinary code called an “ID” conjointly contains a “NAME” and a “DESCRIPTION”. This table may be a central input for data across diverse ventures. It is likely associated with other tables that have particular extended details. The “STATUS” table shows the distinctive stages or states that issues can be in, like “Open”, “In Progress”, or “Closed”. It includes a code and a title for each state. This makes a difference the “ISSUE” table keeps track of how errands are advancing.

The ERD has more tables, each with its claim critical work within the database format. The “Issue Type” sorts issues into distinctive categories like bugs, highlights, and so on. The “Issue Link” interfaces distinctive issues together, showing how they depend on each other. “Issue Watcher” keeps track of individuals who are keeping an eye on or are fascinated by particular subjects. “Priority” implies how critical or imperative something is. “Resolution” implies the conceivable results for an issue, like settling it or not settling it. “Version” appears as distinctive shapes of an aspect or portion. A “Security Scheme” decides who can see or alter certain things, based on their level of getting to.

These tables are closely associated with utilizing keys, which makes a difference make a web of information that underpins a complex extended administration framework. The way information is organized makes a difference and beyond any doubt, it is exact and makes it simpler to discover data. This permits individuals to analyze the information and make shrewd choices approximately ventures and issues within the organization.

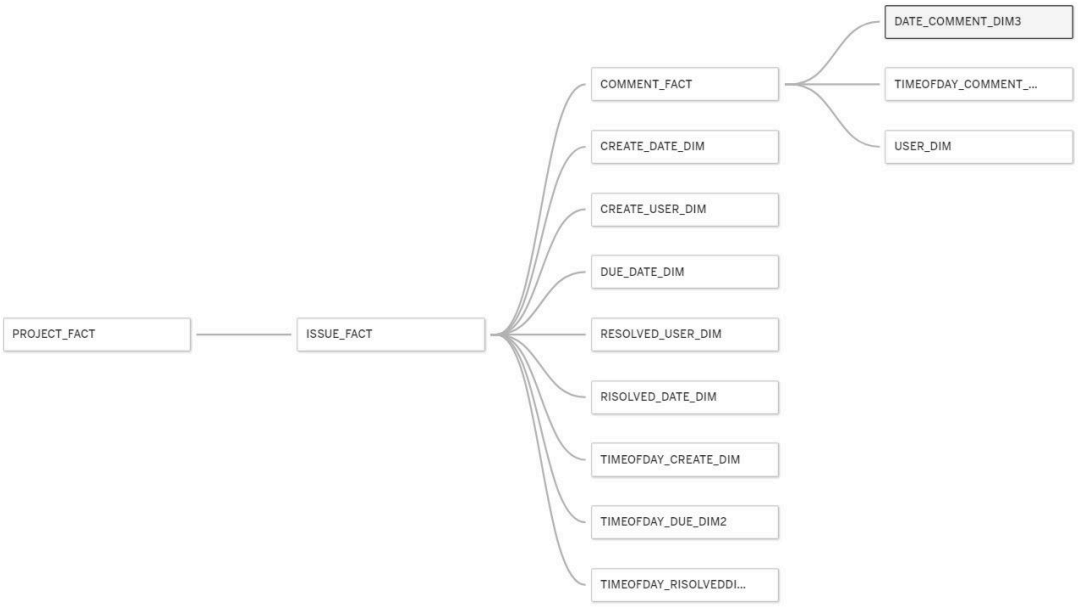
The database pattern delineated within the ERD gives a vigorous system for an application likely centered on extended administration, following, and collaboration. The clearly characterized connections, meant by the utilization of remote keys, permit information from different tables to be interconnected coherently, advertising a consistent stream of data all through the application. The database plan illustrates a mindful approach to capturing the complex nature of venture administration assignments and client intuition inside a framework. This structure not as it were encourages information recovery and control but also guarantees information keenness and a strong establishment for client interface intuitive, analytics, and detailing capacities. Through this well-organized pattern, the database is well-equipped to be the spine of an energetic application that caters to a huge number of extended administration necessities and client workflows.

Dimensional Modeling

Dimensional modeling is essential for transforming raw data into meaningful analytics aligned with our project's strategic objectives. Our dimensional model is proposed to address pivotal aspects of project management, by identifying the creators and solvers of issues, associating issues with their respective projects, and tracking issue creation, updating, and resolution dates; in the meantime, it also captures the comments through the workflow process.

We constructed the following tables with three main criteria:

- We wanted each row to be unique with some KPIs being included in the fact tables
- Each table addresses one of the who, where, when and what questions
- The goal is to create tables that can allow the creation of KPIs by only joining a few tables together



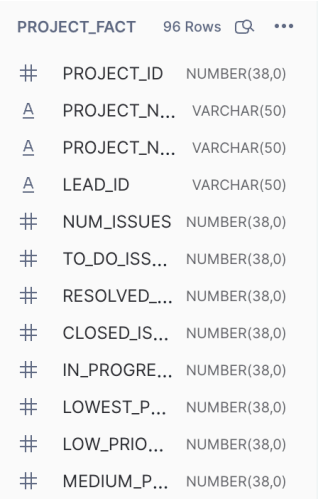
Based on the modeling schema above, we create the following fact and dimension tables.

Fact Tables:

- ISSUE_FACT: tracks individual issues with issue ID, project ID, create user, resolved user, create day and time, resolve date and time, and due date and time.

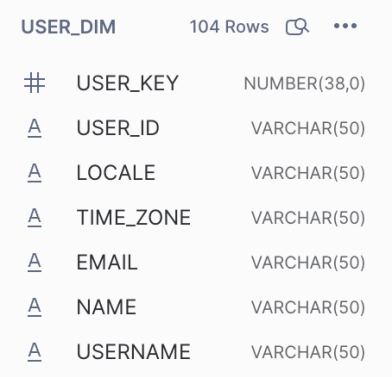
- COMMENT_FACT: stores comments made on issues, containing comment ID, issue ID, author user, and comment date and time.

- PROJECT_FACT: holds information about the project, including project ID, project name abbreviation, project name, lead ID, number of issues, to-do issues, resolved issues, closed issues, in-progress issues, and sorted by priorities of the issues.

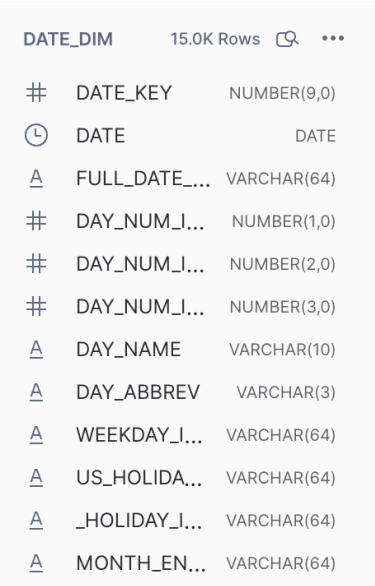


Dimension Tables:

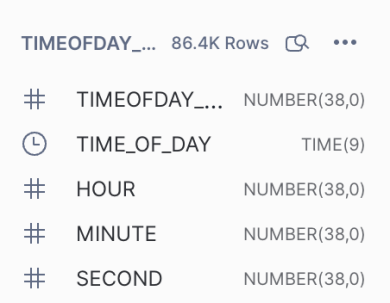
- USER_DIM: contains user information including user ID, location, time zone, and name.



- DATE_DIM: contains timeframe based on dates.



- TIMEOFDAY_DIM: contains timestamps based on the time of the day.



These dimensions are chosen to capture the most relevant aspects of project management for more efficient and effective analysis. We decided to create a comment fact and project fact to avoid too many relationships between the tables. Each issue can be commented on multiple times before its completion and projects are usually assigned multiple issues.

A similar argument can be made for dates and times of day. In fact, each issue is created, resolved, commented and closed at different dates and times and needs to be completed before the due date. In order to keep unique rows in both the issue table and the date and time of day dimensions, we included different date and time keys in the issue table, each one for the different events (creation, completion, closure, etc), that can be joined back with the corresponding date and time of day keys in the date and time of day dimension tables. We ensured unique identifiers from each table in order to maintain data integrity.

The project fact could easily have been a project dimension, but since we wanted to include KPIs for each row (corresponding to a project) we decided to make it into a fact table.

Data Transformation

Most of the data we needed to compute KPIs was not missing. The only variable that contained a large number of null values was due_date. This variable is contained in the original issue table and shows the deadline to complete an issue. It was necessary to compute KPIs that included the completion of issues by the deadline established.

However, after researching how Jira is used to organize projects we discovered that on the platform for each issue users create sprints. When issues (tasks) are already subtasks of another issue, multiple sprints are created to further divide tasks into smaller assignments, each one having a different sequential due date. Therefore we decided that we could use the last due date of the last sprint assigned to an issue to represent the final due date of the issue. We then joined the sprints table to the issue table and used coalesce to select the first non-null value between the original due_date column and the latest due date of the sprint.

Produce KPIs using Fact and Dim

The first KPIs are project-related and measure the number of issues created, resolved and closed per project. They can be an indication of the usage of Jira to manage the organization of projects and give an overall view of the project's completion. If the number of issues created per project is larger than once resolved and issues are closed for lack of a solution, then there could be project completion inefficiencies. This could also be an indication of areas of improvement if issues are not resolved across projects in the company.

Number of issues created per project

To create this KPI using our fact and dimension tables we only needed to join the project fact and the issue fact table. Each issue in the fact table is a row in the table and is connected to the project fact table through the project ID. Each issue in the table is by default created; computing the count(issue_id) after grouping by project_id yields the number of issues created per project.

```
select
COUNT(issue_id) as number_of_issues_created_per_project,
project_id
from issue_fact
group by project_id;
```

NUMBER_OF_ISSUES_CREATED_PER_PROJECT	PROJECT_ID
772	10002
7	10036
126	10004
56	10015
36	10096
135	10049
65	10042
79	10034
262	10013
33	10100
12	10014

Number of issues created, resolved, and closed per project

The query to extract the number of issues resolved and closed is similar to those created. The issue contains a resolved flag and a closed flag. After grouping by project_id a SUM of the respective flags will yield the KPIs needed. With all three project-specific KPIs it is possible to track easily the workflow for each project.

```
select
COUNT(issue_id) as number_of_issues_created_per_project,
SUM(resolved_flag) as number_of_issues_resolved_per_project,
SUM(closed_flag) as number_of_issues_closed_per_project,
project_id
from issue_fact
group by project_id;
```

NUMBER_OF_ISSUES_CREATED_PER_PROJECT	NUMBER_OF_ISSUES_RESOLVED_PER_PROJECT	...	NUMBER_OF_ISSUES_CLOSED_PER_PROJECT	PROJECT_ID
772	483		0	10002
569	525		0	10001
309	177		0	10059
262	161		0	10013
135	94		0	10049
132	101		0	10008
126	86		0	10004
102	45		0	10011
94	84		0	10037
86	73		0	10018
79	65		0	10034

User Activity: This KPI can track various metrics related to individual user activity, such as the number of issues created, resolved, or commented on. This helps us identify top performers and areas for improvement. Moreover, the Project Completion Rate measures the percentage of projects that are completed on the estimated time

User activity

This user activity KPI can track various metrics related to individual user activity, such as the number of issues created, resolved, or commented on. This helps us identify top performers and areas for improvement. Moreover, the project completion rate measures the percentage of projects that are completed in the estimated time. User activity is important on many different levels. The number of issues created by a user is useful to gather information on the performance in the company of an employee. Even though the creator of an issue oftentimes assigns an issue to a coworker, and thus does not resolve the issue themself, the personal activity on Jira could still be very valuable to the company. First of all, Jira is a paid service that the company invested money on. Furthermore, the company likely believes that Jira could improve the organization and efficiency of projects and might care whether a manager uses the license. The company may also be interested in learning about the success of projects under the management of the creator of the issue. All this can be captured by the number of issues created by the user.

To obtain the creation by user KPI from our fact and dimension table it is necessary to join the issue table to the user_create_dim table. Each issue in the issue has a column recording the create_user_key that corresponds to a user_id in the user_create_dim. Once the two tables are joined and grouped by the user, a count of the issues ID will output the number of issues created by the user.

```
select
u.user_id,
u.name,
COUNT(i.issue_id) number_of_issues_created_by_user
from issue_fact i
inner join user_dim u
on i.create_user_key = u.user_key
group by u.user_id, u.name
;
```

USER_ID	NAME	NUMBER_OF_ISSUES_CREATED_BY_USER
5e8d2bee2c0eff0b8fb3e98c	Melissa Alvarez	1218
5e9104e53a90600b96989170	Franck Leveneur	149
630d246656010c40d44632f9	Emilio Paez	228
61b0c375d5986c006a9eddd4	Kamaljeet Kaur	428
63f55b2989de3d475af4c5a1	Serenity Shields	76
6179d16d860f78006b43f7cd	Kevin Ding	18
712020:8fd9fa78-e542-4aa1-9aca-fb0211e5dde3	Haitong Huang	5
60342bb425b84e00692dda7d	Andy	36
613fba41805a97006a9b04ed	Grant Huber	171
61fc2c8fb3ec760068c26ae8	Ehsan Hemati	10
60dd0b8f285656006a877a9e	Riti Chrea	17
62f6598df15eecaf5010e1c0	Ovais Siddiqui	90
70121:5fff1c96-f765-4dc4-af97-930ea853cc91	Jay Chen	4
5ed56e586d27410c1e25f467	Ranjan Srinivas	9

The KPI measuring the number of issues resolved has a very clear objective: it highlights the top performers in the company. As long as the company can make sure that the credit is given to the right employee (in our case we can only see who the issue was assigned to), the KPI will be of relevance to monitor the business.

The KPI can be retrieved from the fact and dimension tables similar to the number of created issues per user. After joining the issue fact with the resolved_user_dim on the resolved_user_key and grouping by user_id, a count of the issue_ids gives out the number of issues resolved by the user.

```
select
u.user_id,
u.name,
COUNT(i.issue_id) number_of_issues_resolved_by_user
from issue_fact i
inner join user_dim u
on i.resolved_user_key = u.user_key
group by u.user_id, u.name
;
```

USER_ID	NAME	NUMBER_OF_ISSUES_RESOLVED_BY_USER
5a743fcdf15e7f2cc08afed6	Игорь Черненко	88
5e9104e53a90600b96989170	Franck Leveneur	323
62100e6fbba9ca0070cca27a	Suhas Sridhar	7
61973f59c510bc006b4af9ec	Pranit Bhaskar Sherkar	23
6179d16d860f78006b43f7cd	Kevin Ding	44
6111d697627b56006869940e	bala	234
63596697fe5ff375235a25a8	Jas Mowgood	39
60727d0fc642ff0070b4fed9	James Darrel Bautista	33
610c00100b454a00682eace8	Subodh Kalika	40
5ed56e586d27410c1e25f467	Ranjan Srinivas	36

Number of issues commented on by user

This KPI could be a proxy for the involvement and teamwork of a user in a project. Comments are usually utilized to record an update on the issue or to highlight what needs to be done.

To compute the KPI we joined the comment fact table with the user dim table. The tables are joined on the user key and once the tables are grouped by user ID and name the count of comment IDs results in the total number of comments per user.

```
select
COUNT(c.comment_id) as number_of_comments_per_user,
u.user_id,
u.name
from comment_fact c
inner join user_dim u
on c.author_user_key = u.user_key
group by u.user_id, u.name;
```

NUMBER_OF_COMMENTS_PER_USER	USER_ID	NAME
3061	5e8d2bee2c0eff0b8fb3e98c	Melissa Alvarez
1175	5e9104e53a90600b96989170	Franck Leveneur
736	70121:a47e57dc-b3a5-4fd5-ac13-e5f43efc5878	Stavros Papadakis
291	61b0c375d5986c006a9eddd4	Kamaljeet Kaur
216	630e9ac962fe1e6eac6c22dc	Deborah Arellano
192	61dfd00049f1950069b27d6e	Alex Yarosh
174	63596697fe5ff375235a25a8	Jas Mowgood
160	60727d0fc642ff0070b4fed9	James Darrel Bautista
157	6111d697627b56006869940e	bala
138	712020:6db7d9f4-b9a6-4f46-932b-e0dcdb91bc11	Rateeb Yehya
118	712020:0f968d0c-ee68-457b-b31d-ad5fb802c537	GI Griffin
118	712020:c678f7ad-d748-495e-8018-883e3692e9fe	Anima V V
113	60351e604623c60069adf39b	Antun

Percentage of projects completed in the estimated time

In the original table for projects, there is no indication of project completion. We therefore decided to compute the project completion rate based on the number of issues that were resolved out of the issues composing the project. The rate of completion is very low, with only 9 projects being completed. The number of projects completed before the due date is 0, as in no projects all issues were resolved before the due date.

However, it is still possible to compute the number of projects completed before the due date through the project fact. Each row is a project and KPIs that are related to the project are included in the columns. Examples are the number of issues completed before the due date, the number of issues, the number of high-priority issues, etc.

```
select project_id, project_name,PERC_RESOLVED_BEFORE_DUE_DATE_ISSUES
from mydb.jira_analytics.project_fact;
```

This query generates the percentage of issues per project that were resolved before the due date. Only 7 projects had some of their issues completed in time, suggesting that either due dates are set wrongly or issues are rarely resolved before the due date.

PROJECT_ID	PROJECT_NAME	PERC_RESOLVED_BEFORE_DUE_DATE_ISSUES
10025	True Dialog - Voice Shot	71.93
10004	Hyperwolf	68.25
10028	UR Labs	68.18
10040	AdAction	64.10
10036	EZClocker	57.14
10045	International Relief Teams	53.70
10046	Betwext	40.00
10076	SuperHote	0.00
10073	Sportivoai	0.00
10071	Customily Inc	0.00
10003	CMX	0.00

Cycle time

This KPI measures the average time it takes to complete an issue (ticket), from creation to resolution. This can help you identify bottlenecks and improve your workflow efficiency.

This KPI is calculated by using the issue fact table to get the creation and resolution date key for each issue (for resolved issues) and then joined with the date and time of day dimensions to get the exact timestamps corresponding to the creation and resolution of an issue. The cycle time was calculated by finding the time taken to resolve an issue from the time of its creation. It was calculated at different levels (seconds, minutes, hours, and days). The KPI was calculated individually for each issue as well as an aggregated metric calculating the average cycle time.

```
SELECT ISSUE_ID,
DATEDIFF(SECOND, CREATION_DATETIME, RESOLUTION_DATETIME) AS CYCLE_TIME_SECS,
DATEDIFF(MINUTE, CREATION_DATETIME, RESOLUTION_DATETIME) AS CYCLE_TIME_MINS,
DATEDIFF(HOUR, CREATION_DATETIME, RESOLUTION_DATETIME) AS CYCLE_TIME_HRS,
DATEDIFF(DAY, CREATION_DATETIME, RESOLUTION_DATETIME) AS CYCLE_TIME_DAYS
FROM
    (
    SELECT A.ISSUE_ID,
    TIMESTAMP_NTZ_FROM_PARTS(B.DATE::DATE, C.TIME_OF_DAY::TIME) AS CREATION_DATETIME,
    TIMESTAMP_NTZ_FROM_PARTS(D.DATE::DATE, E.TIME_OF_DAY::TIME) AS RESOLUTION_DATETIME
    FROM MYDB.JIRA_ANALYTICS.ISSUE_FACT A
    INNER JOIN MYDB.JIRA_ANALYTICS.DATE_DIM B
    ON A.CREATE_DATE_KEY = B.DATE_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.TIMEOFDAY_DIM C
    ON A.CREATE_TIME_KEY = C.TIMEOFDAY_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.DATE_DIM D
    ON A.RESOLVED_DATE_KEY = D.DATE_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.TIMEOFDAY_DIM E
    ON A.RESOLVED_TIME_KEY = E.TIMEOFDAY_KEY
    WHERE A.RESOLVED_FLAG = 1
    ) A ;
```

	ISSUE_ID	···	CYCLE_TIME_SECS	CYCLE_TIME_MINS	CYCLE_TIME_HRS	CYCLE_TIME_DAYS
1	10982		9383603	156393	2606	108
2	11658		1603883	26731	445	18
3	11471		32283437	538057	8967	373
4	11472		8093781	134896	2248	93
5	11473		6718182	111970	1866	77
6	11398		1049152	17486	291	1
7	13457		1119065	18651	310	12
8	11474		7494276	124905	2081	86
9	11475		12161268	202688	3378	140
10	11476		8486129	141435	2357	98

```
SELECT
AVG(DATEDIFF(SECOND, CREATION_DATETIME, RESOLUTION_DATETIME)) AS AVG_CYCLE_TIME_SECS,
AVG(DATEDIFF(MINUTE, CREATION_DATETIME, RESOLUTION_DATETIME)) AS AVG_CYCLE_TIME_MINS,
AVG(DATEDIFF(HOUR, CREATION_DATETIME, RESOLUTION_DATETIME)) AS AVG_CYCLE_TIME_HRS,
AVG(DATEDIFF(DAY, CREATION_DATETIME, RESOLUTION_DATETIME)) AS AVG_CYCLE_TIME_DAYS
FROM
    (
    SELECT A.ISSUE_ID,
    TIMESTAMP_NTZ_FROM_PARTS(B.DATE::DATE, C.TIME_OF_DAY::TIME) AS CREATION_DATETIME,
    TIMESTAMP_NTZ_FROM_PARTS(D.DATE::DATE, E.TIME_OF_DAY::TIME) AS RESOLUTION_DATETIME
    FROM MYDB.JIRA_ANALYTICS.ISSUE_FACT A
    INNER JOIN MYDB.JIRA_ANALYTICS.DATE_DIM B
    ON A.CREATE_DATE_KEY = B.DATE_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.TIMEOFDAY_DIM C
    ON A.CREATE_TIME_KEY = C.TIMEOFDAY_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.DATE_DIM D
    ON A.RESOLVED_DATE_KEY = D.DATE_KEY
    INNER JOIN MYDB.JIRA_ANALYTICS.TIMEOFDAY_DIM E
    ON A.RESOLVED_TIME_KEY = E.TIMEOFDAY_KEY
    WHERE A.RESOLVED_FLAG = 1
    ) A ;
```

	AVG_CYCLE_TIME_SECS	AVG_CYCLE_TIME_MINS	···	AVG_CYCLE_TIME_HRS	AVG_CYCLE_TIME_DAYS
1	3705663.319660	61761.049889		1029.356615	42.924982

Project progress and status based on issue statuses, priorities, and due dates

This KPI provides a snapshot of the overall health of projects by tracking the status of issues, their priorities, and their due dates. It contains multiple details about the project based on the different types of issues associated with it and their status.

This KPI was determined using the project fact table. It contains information at a project level, such as the # of issues associated with a project, and their breakdown depending on their status: to-do issues, resolved issues, and in-progress issues. The issues are also broken down based on their priority into 5 categories: Lower, Low, Medium, High and Higher. For each of

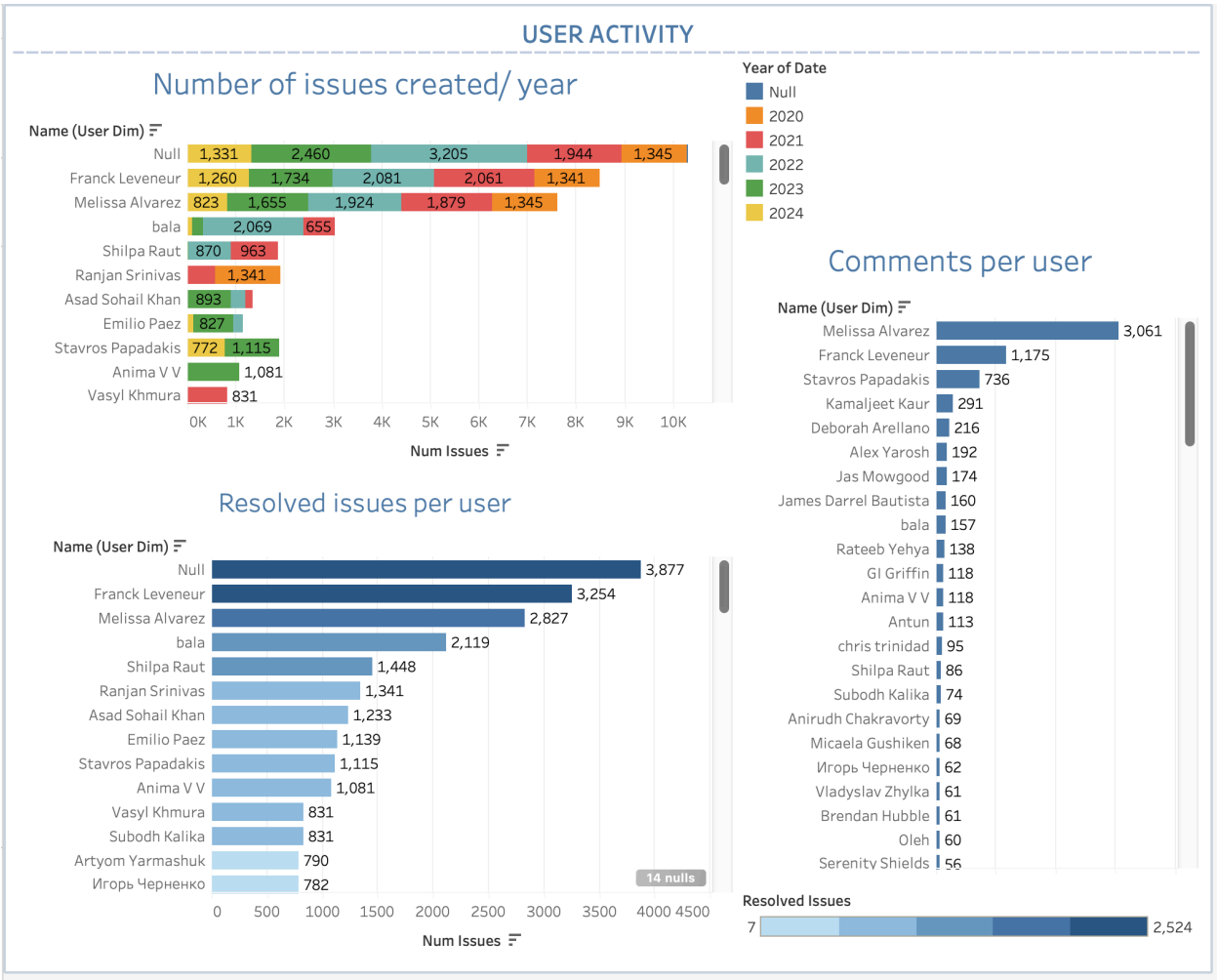
these 5 priority categories, the % of issues that have been resolved have been added. The % of issues that were completed before their due date has also been calculated.

```
SELECT PROJECT_ID, PROJECT_NAME,
NUM_ISSUES, TO_DO_ISSUES, RESOLVED_ISSUES, IN_PROGRESS_ISSUES,
LOWEST_PRIORITY_ISSUES, LOW_PRIORITY_ISSUES, MEDIUM_PRIORITY_ISSUES, HIGH_PRIORITY_ISSUES, HIGHEST_PRIORITY_ISSUES,
RESOLVED_BEFORE_DUE_DATE_ISSUES, PERC_TO_DO_ISSUES, PERC_RESOLVED_ISSUES, PERC_IN_PROGRESS_ISSUES,
PERC_LOWEST_PRIORITY_ISSUES, PERC_LOW_PRIORITY_ISSUES, PERC_MEDIUM_PRIORITY_ISSUES,
PERC_HIGH_PRIORITY_ISSUES, PERC_HIGHEST_PRIORITY_ISSUES, PERC_LOWEST_PRIORITY_ISSUES_DONE,
PERC_LOW_PRIORITY_ISSUES_DONE,  PERC_MEDIUM_PRIORITY_ISSUES_DONE, PERC_HIGH_PRIORITY_ISSUES_DONE,
PERC_HIGHEST_PRIORITY_ISSUES_DONE, PERC_RESOLVED_BEFORE_DUE_DATE_ISSUES
FROM MYDB.JIRA_ANALYTICS.PROJECT_FACT;
```

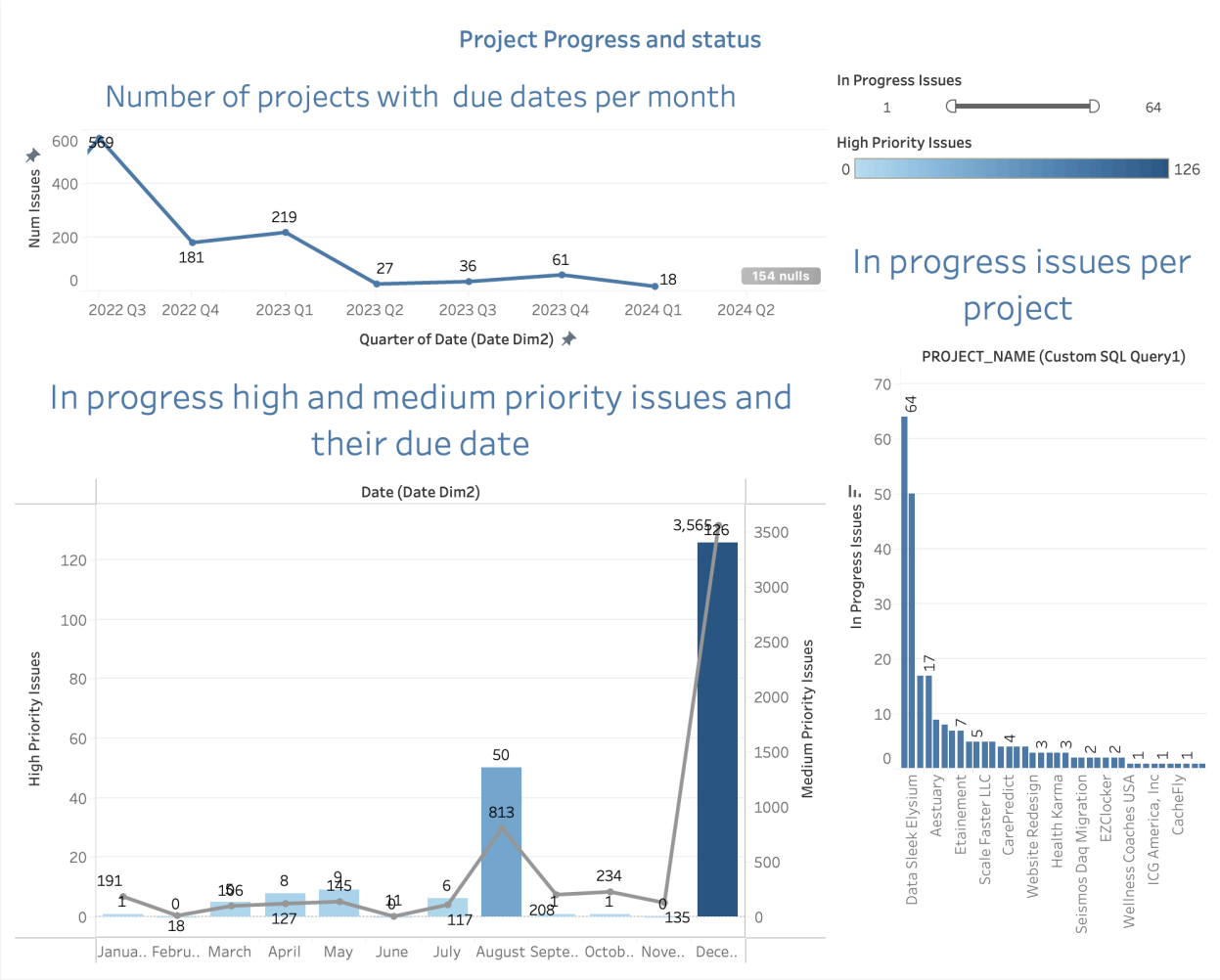
	PROJECT_ID	PROJECT_NAME	NUM_ISSUES	TO_DO_ISSUES	RESOLVED_ISSUES	IN_PROGRESS_ISSUES	···	LOWEST_PRIORITY_ISSUES	LOW_PRIORITY_ISSUES	MEDIUM_PRIORITY_ISSUES
1	10076	SuperHote	0	0	0	0		0	0	0
2	10073	Sportivoai	0	0	0	0		0	0	0
3	10071	Customily Inc	0	0	0	0		0	0	0
4	10003	CMX	35	24	2	3		0	0	35
5	10005	Health Karma	71	7	43	3		0	0	69
6	10006	North Labs	5	1	4	0		0	0	5
7	10007	Flexivan	12	1	11	0		0	0	12
8	10008	Mirai	132	17	101	8		0	1	131
9	10011	Auto Rescue Solutions	102	37	22	17		0	0	101
10	10012	Reveneer	0	0	0	0		0	0	0
11	10014	Livestock Nutrition	12	1	10	0		0	0	12
12	10016	Omure	6	2	4	0		0	0	6
13	10017	Fruitful Source	42	28	0	2		0	0	39
14	10018	Digital Asset Research	86	2	73	2		0	0	86
15	10019	Rasa.io	5	0	5	0		0	0	5
16	10020	Act Now Technologies	2	0	2	0		0	0	2
17	10023	Shearer Supply Inc	10	3	1	5		0	0	10
18	10024	Etainement	46	9	30	7		0	0	45

Data Visualization

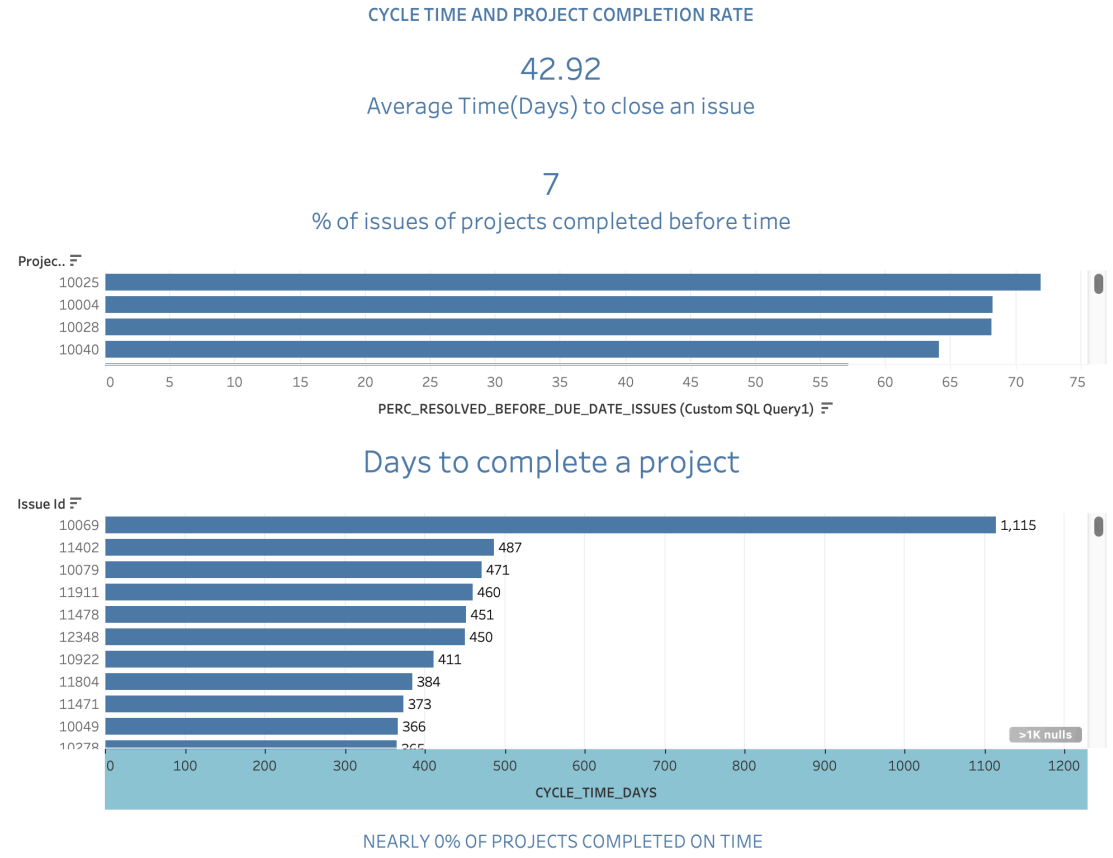
The data visualization part was entirely made in Tableau with a little help from the SQL queries that were executed. This data provides further insight into what should be improved on and how to proceed with accessing this data. A thing to consider when looking at the data is that the data we accessed has SQL queries in the visualization. Note: we chose to submit 4 dashboards instead of 5 as there is sufficient data for the cycle time KPI and Completion rate, which we uncovered earlier in the SQL query.



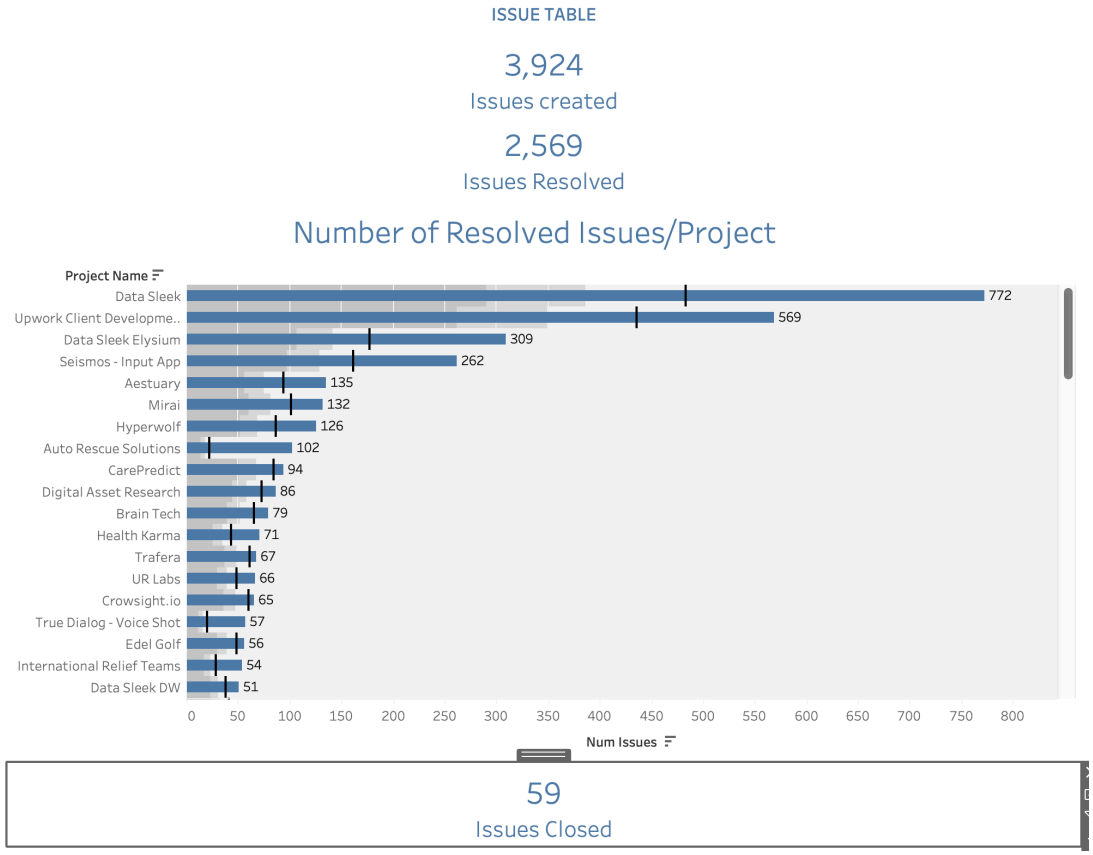
In the first dashboard, User Activity, we uncovered who our top performers were in terms of issue creating/resolving and commenting. By looking at this data, it is evident that there are several high achievers who are most active in working on the projects. By having this KPI, it is much easier to target certain users that would be beneficial for moving projects forward. Another thing to distinguish here is that these users have varying activity over different years.



The second dashboard includes the KPIs of project progress and status. We chose to include this as it is the most important for projects that are about to be completed/closed. It is evident that December has the highest amount of projects due in the high/medium priority, so this metric should show that the team must work towards completing that goal. Another interesting thing to consider here is that there are only several projects that have in-progress issues due soon. By looking at this data, the company might decide to close out a certain project first by completing all of its issues. Moreover, there is also the indicator of how many projects are due during which quarter.



This dashboard includes the average time to complete an issue (almost 43!) which means that there is too much time spent on every issue and even some of the issues take more than a year (and in 1 case more than 3 years) to complete. Looking at this, management should consider either hiring more people to complete projects or finding a way to streamline their issue-solving. Another big fact that we found is that only 7% of issues of projects are completed before the deadline, meaning that time really is short when doing these issues. An important caveat is that nearly 0% of projects are completed on time, meaning that there really is a shortage of people/ lack of good time planning for these issues of projects.



The Last dashboard includes the issue table- namely how many issues are created/resolved and closed. The number of closed issues is only 59 out of the 3924 that have been created which gives a low closed rate. Another thing to consider here is the amount of issues created by Jira's biggest clients. It is evident that they have a lot more issues than everyone else. Moreover, additional help should be required there. It is evident that issues are prioritized by deadline, instead of by project as most of them have some sort of completion.

Project Challenges

One of our challenges was managing high-pressure and busy schedules, especially during the final week. However, we successfully maintained productivity and stayed focused on our project objectives. Although finding time outside of class was difficult, we made efficient use of classroom hours and messages for coordination and progress tracking. Despite the time limitations, we ensured that all tasks were completed with thoroughness and precision.

The biggest challenge was accessing all 31 different tables and joining them on the correct keys. As the dataset had a huge variety of tables/sources we had to make sure that everything was done correctly when setting up these keys.

The Tableau part was particularly challenging because it required the data to be accessed from different dates/timetables. The varying essence of the comment, due date, completion, and creation of the issues is difficult to access by itself. Moreover, looking at the different data sources, we had to join the data on different keys and visualize it carefully without making mistakes in choosing a table. It did require a custom SQL script to run more smoothly, so we had to be careful when choosing the data.