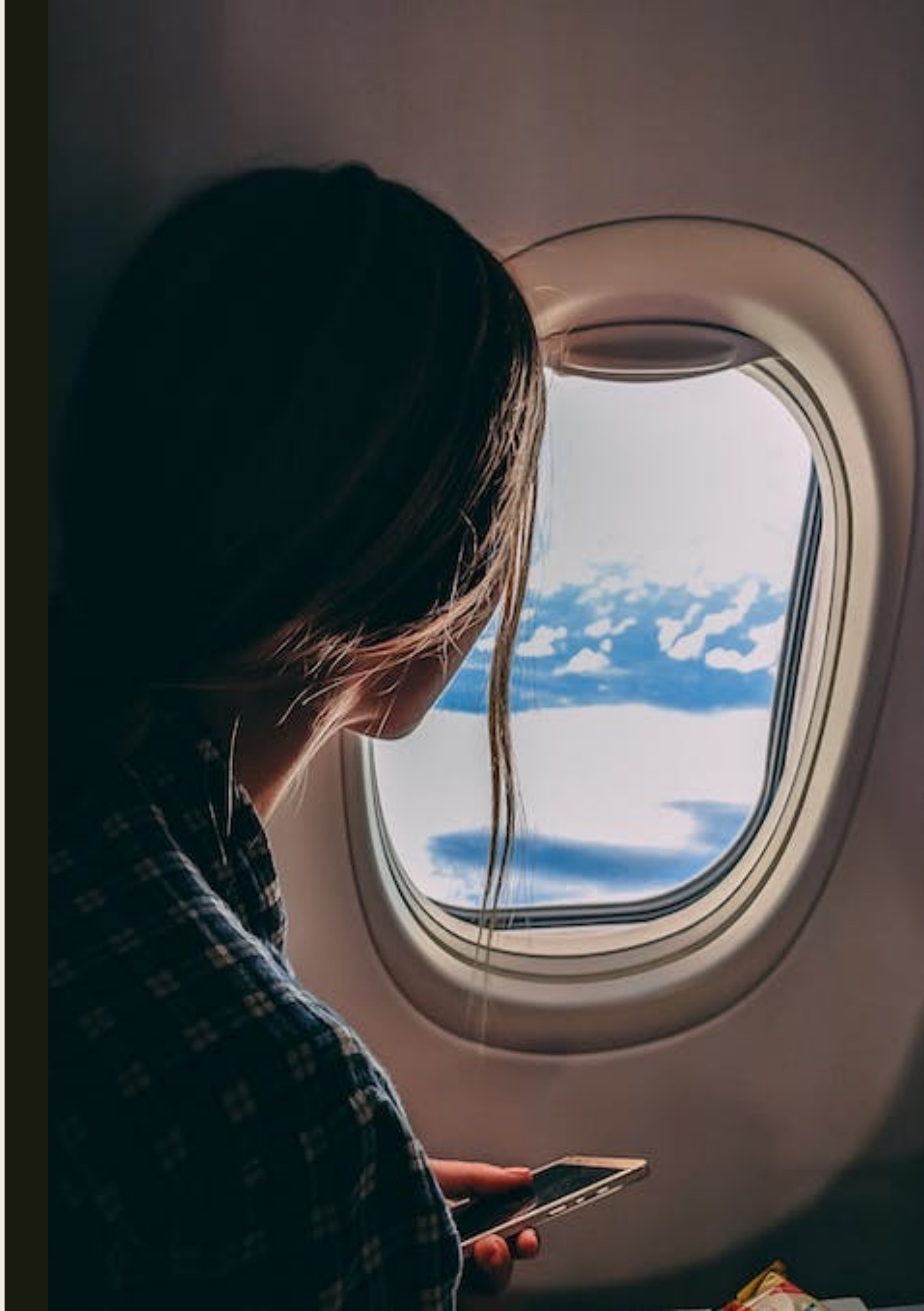
TRAVEL SLATE

By: David Giacobbi

Project Overview

- Travel Slate is a database application intended to make planning vacations easier
- Users can create an account where all potential travel plans can be stored in a library
- Users also can write reviews on the plans that they have purchased and give them an overall rating
- Users use the application's search engine to perform faceted searches on the library of available plans
- Admin page is also available for Travel Slate to get analytics on their consumer's data



Project Demo

- Login Error Checking
- Creating a New Account
- Adding a plan using faceted search
- Writing a review on plan in user library
- Removing a review and plan
- Admin page of SQL Analytics

ER Diagram

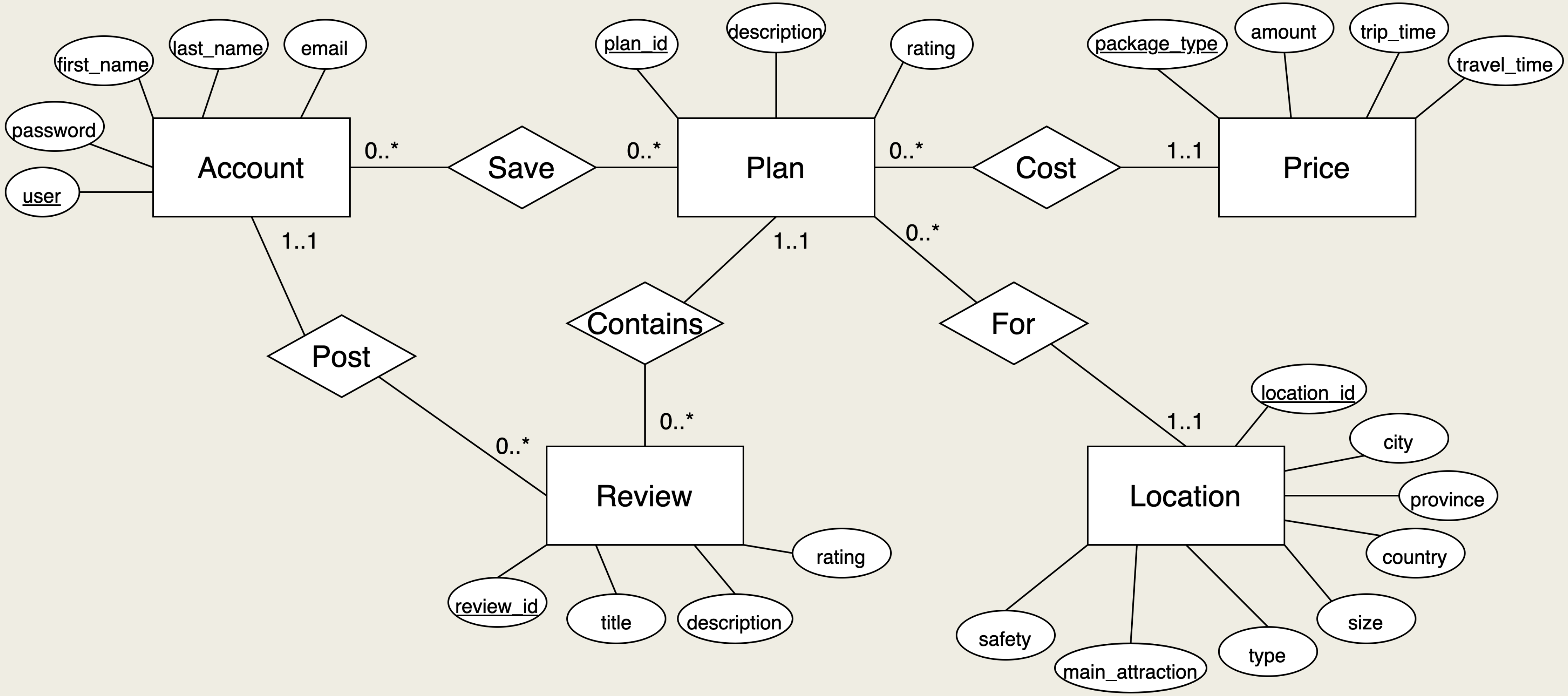

first_name
last_name
email
password
user
Account
0..*
Save
0..*
plan_id
description
rating
Plan
0..*
Cost
1..1
package_type
amount
trip_time
travel_time
Price
1..1
1..1
0..*
Post
Contains
For
0..*
0..*
1..1
location_id
city
province
country
size
Review
rating
review_id
title
description
Location
safety
main_attraction
type

Table Walkthrough

- Account (user, password, first_name, last_name, email)
- Location (location_id, city, province, country, size, type, main_attraction, safety)
- Price (package_type, amount, trip_time, travel_time)
- Plan (plan_id, description, rating, price_package, location_id)
- SavedPlan (user, plan_id)
- Review (review_id, title, description, rating, plan_id, user)

* Foreign Keys connect the above tables

Difficult Query Walkthrough

Plan Query:

```
SELECT pl.description, pl.rating, l.city, l.province, l.country, l.size, l.type, l.main_attraction, l.safety, p.package_type, p.amount, p.trip_time, p.travel_time

FROM Plan pl JOIN Location l USING (location_id)

                    JOIN Price p ON (p.package_type = pl.price_package)

WHERE plan_id = ?;
```

Consumers w/ Lowest Review Rating Average:

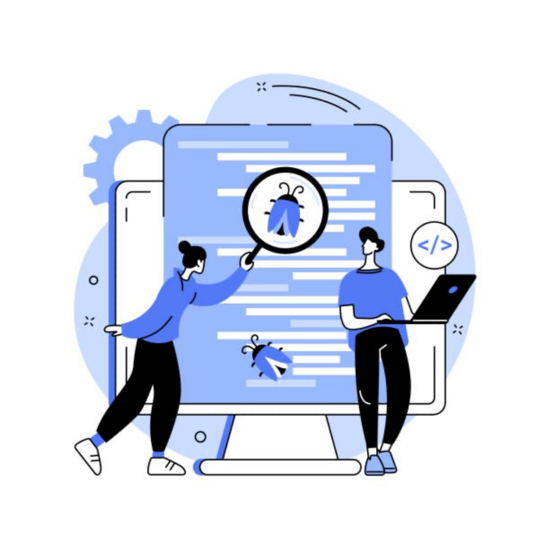
```
SELECT a.user, a.first_name, a.last_name, a.email, ROUND(AVG(r.rating), 2) AS avg_rating

FROM Review r JOIN Account a USING (user)

GROUP BY a.user

ORDER BY avg_rating

LIMIT 5;
```

Plans w/ Lowest Review Rating Average:

```
SELECT p.plan_id, l.city, l.country, p.price_package, ROUND(AVG(r.rating), 2) AS avg_rating

FROM Review r JOIN Plan p USING (plan_id)

                     JOIN Location l USING (location_id)

GROUP BY p.plan_id

ORDER BY avg_rating

LIMIT 5;
```

Application Testing

- **Integration Testing:** used heavily throughout the project to ensure queries were completing and pages were loading/functioning properly
- **User Acceptance Testing:** utilized to briefly check how user friendly the UI design was; it also acted as a second form of functional testing of features implemented
- Database entries were also used as testing as well, especially for reviews and saved plans to ensure that they were working for all users



Challenges/Shortcomings

1. Page Traversal
   - Difficult passing variables through different pages to carry variables
   - Issue with admin page, where it can be accessed despite credential check
2. Special Cases for Error Checking
   - No Filter on all faceted search attributes issue
3. SQL Analytics Testing
   - Data population was difficult to allocate enough to see how effective admin page was at providing insight

Further Application Development

- Have plan ratings automatically update when users add reviews to a certain plan
- Create more attributes to perform faceted search on
- Be able to comment and like other user’s reviews so that they are listed at the top of review tables
- Add more details about travel plans (another table of images or price variation)