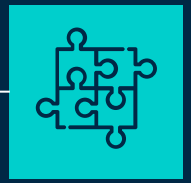
“TICKETDABBLER” DATABASE

Group 3

Arnav, Alisa, Jaya
Nandani, Rebecca

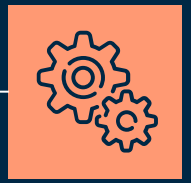
TABLE OF CONTENTS



01

BUSINESS OVERVIEW

- business scenarios
- mission statement & objective



02

DATABASE RELATED

- ERD
- relational schema
- SQL queries

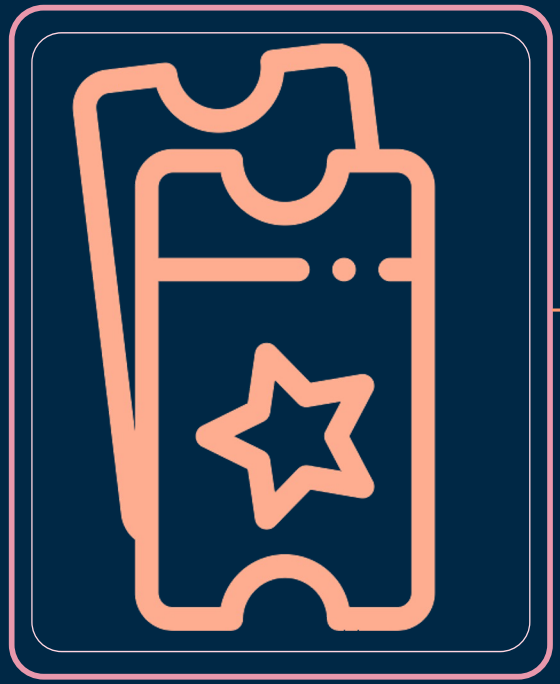
03

LEARNING EXPERIENCES

- challenges
- solutions
- extend the project

OUR BUSINESS

The business is similar to ***Ticketmaster***, a comprehensive event management and ticketing solution platform. With an expansive reach spanning concerts, sports events, theatre productions, and more, our platform connects event clients, venues, and customers in a seamless ecosystem. Customers can explore a diverse range of events, check tickets with ease, and gain access to the events they're passionate about.



MISSION

Statement

The purpose of the Ticketmaster-like database system is to effectively store, manage, and facilitate the exchange of event-related data. Our database fosters seamless collaboration and information sharing among clients, venues, and customers.

Objective

Major Entities: events, venues, address, customers, clients, orders, tickets, seats, reviews

- To maintain (enter, update and delete) data on all entities;
- To perform searches on all entities;
- To report on all entities;
- To track the status of events, events at venues, orders

BUSINESS PROCESS

customers can search events they are interested in according to names, dates, categories, etc

Event Listings

Ticket Purchase

Customers can place orders to buy tickets for events and choose preferred seats

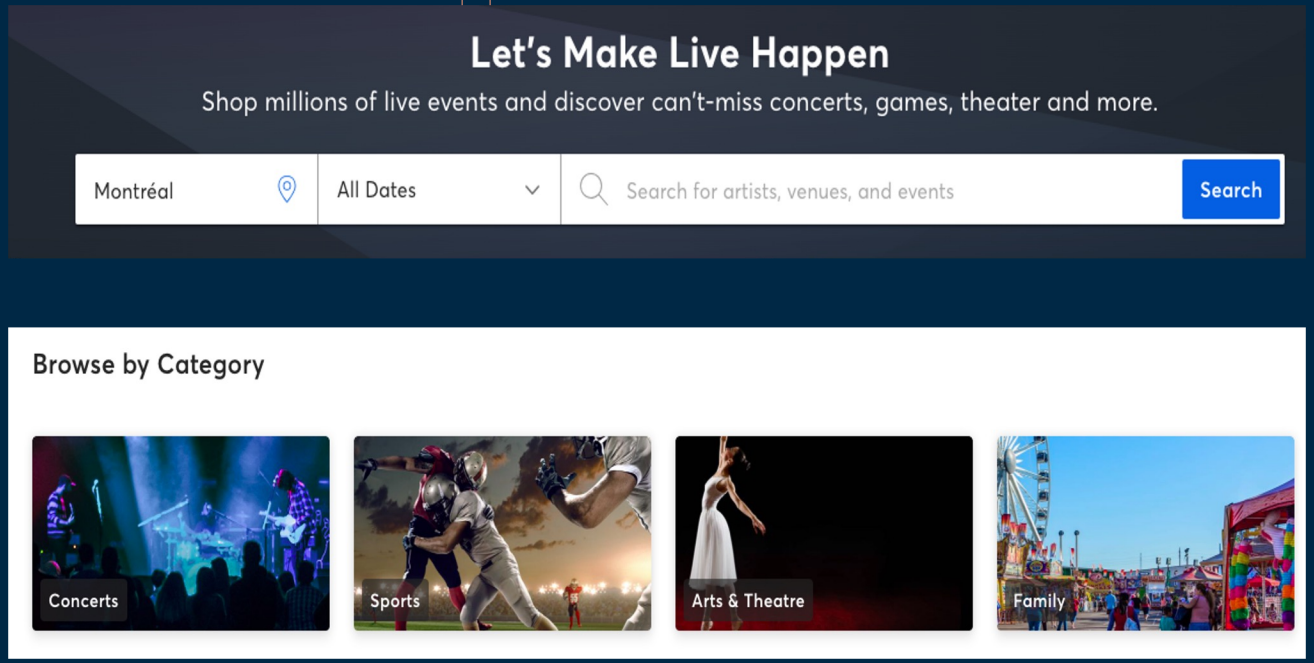
The events are allocated to various venues. The platform records the client information so that they can be contacted

Event Management

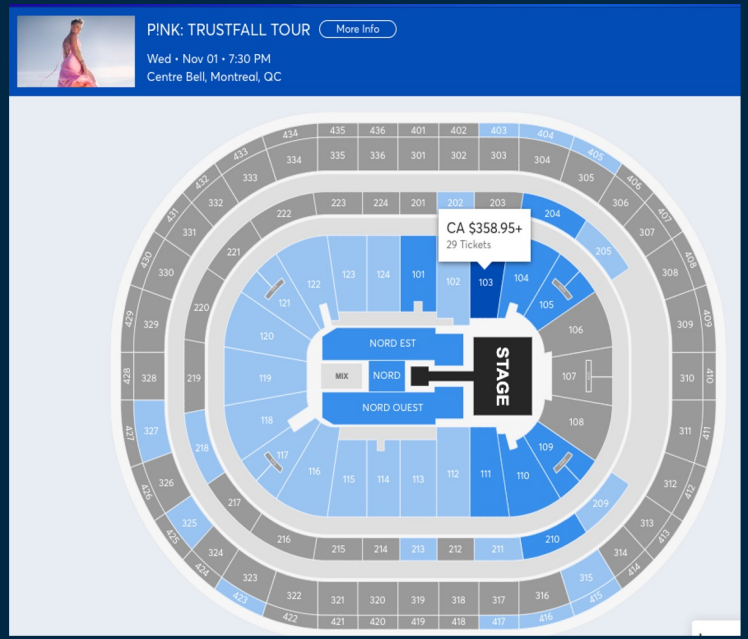
Customer Review

Customers can provide ratings and leave comments for events.

BUSINESS FUNCTIONS: EVENTS SEARCH



BUSINESS FUNCTIONS: TICKETS PURCHASE

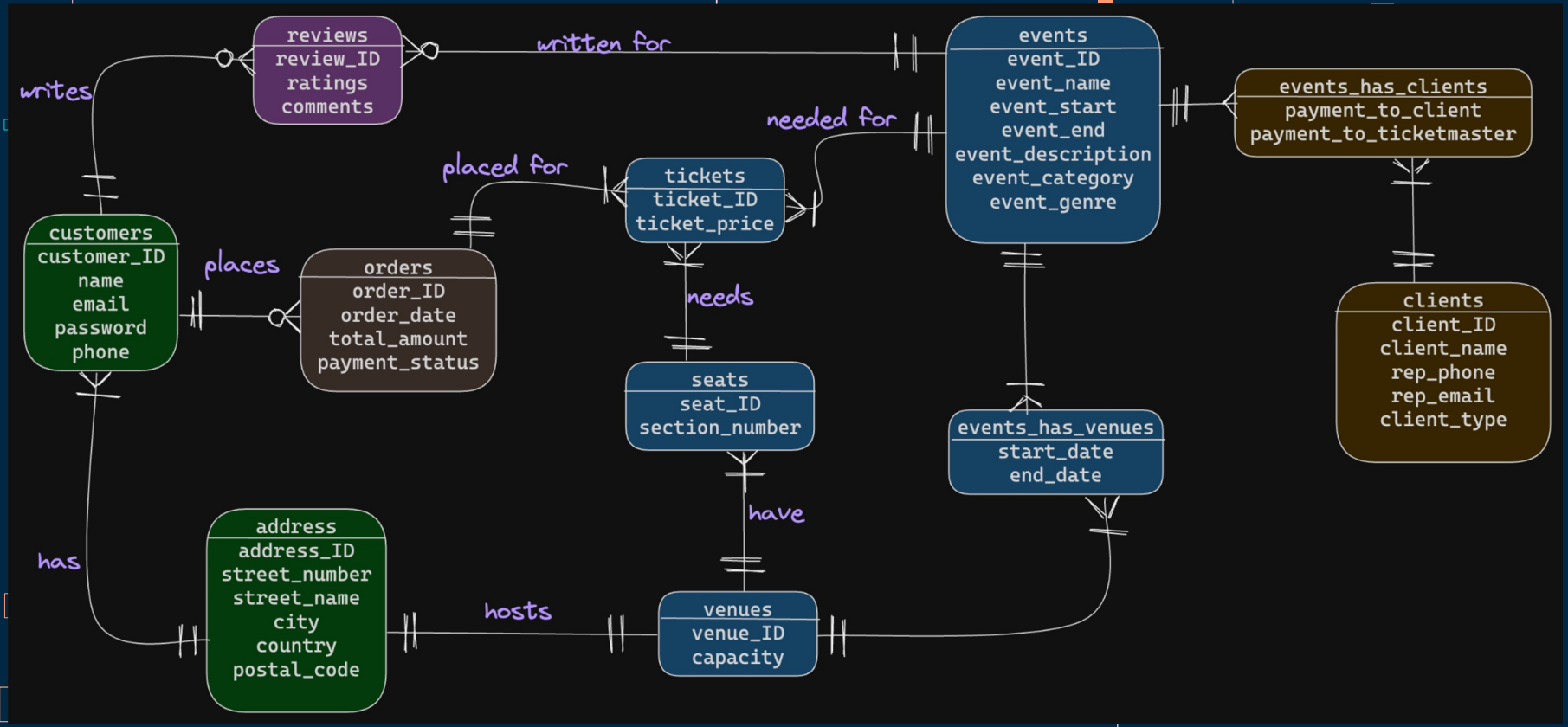


BUSINESS RULES & ASSUMPTIONS

- **Business Rules**:
  - A customer must be associated with an address.
  - An event must have at least one venue and can have multiple venues.
  - An event can have multiple reviews from different customers.
  - An order must be associated with a customer, and each order can have multiple tickets.
  - A ticket must be associated with an event, a seat, and an order.
  - Clients must have events associated with them, and each event must have payments to the client and Ticketmaster.
  - Events must have clients, indicating partnerships between clients and events.
  - Events must have a start and end date, and venues must host events during specific time periods.

- **Business Assumptions**:
  - Customers have a single address associated with them (billing address), assuming customers cannot have multiple addresses. But multiple customers can have the same address.
  - The system assumes that events can take place in multiple venues and across different categories and genres.
  - Ticket prices are stored at the ticket level, assuming they remain constant regardless of the seat or order.
  - Clients are assumed to be external entities or event organizers who collaborate with the event management system.
  - The system assumes that events, orders, and tickets are linked together accurately for tracking purposes.
  - Reviews are associated with customers and events, assuming customers can only leave reviews for events they attended.

ERD


reviews
review_ID
ratings
comments
written for
events
event_ID
event_name
event_start
event_end
event_description
event_category
event_genre
events_has_clients
payment_to_client
payment_to_ticketmaster
writes
needed for
placed for
tickets
ticket_ID
ticket_price
customers
customer_ID
name
email
password
phone
places
orders
order_ID
order_date
total_amount
payment_status
needs
clients
client_ID
client_name
rep_phone
rep_email
client_type
seats
seat_ID
section_number
events_has_venues
start_date
end_date
have
has
address
address_ID
street_number
street_name
city
country
postal_code
hosts
venues
venue_ID
capacity

RELATIONAL SCHEMA

address(address_ID, street_number, street_name, city, country, postal_code)
Primary Key: address_ID

customers(customer_ID, name, email, password, phone, address_ID)
Primary Key: customer_ID
Foreign Key: address_ID References address(address_ID)

events(event_ID, event_name, event_start, event_description, event_end, event_category, event_genre)
Primary Key: event_ID

reviews(review_ID, rating, comments, customer_ID, event_ID)
Primary Key: review_ID
Foreign Key: customer_ID References customers(customer_ID)
Foreign Key: event_ID References events(event_ID)

orders(order_ID, order_date, total_amount, payment_status, customer_ID)
Primary Key: order_ID
Foreign Key: customer_ID References customers(customer_ID)

clients(client_ID, client_name, rep_phone, rep_email, client_type)
Primary Key: client_ID

RELATIONAL SCHEMA CONT.

venues(venue_ID, name, capacity, address_ID)

Primary Key: venue_ID
Foreign Key: address_ID References address(address_ID)

seats(seat_ID, section_number, venue_ID)

Primary Key: seat_ID
Foreign Key: venue_ID References venues(venue_ID)

tickets(ticket_ID, ticket_price, events_ID, order_ID, seat_ID)

Primary Key: ticket_ID
Foreign Key: event_ID References events(event_ID)
Foreign Key: order_ID References orders(order_ID)
Foreign Key: seat_ID References seats(seat_ID)

events_has_clients(event_ID, client_ID, payment_to_client, payment_to_ticketmaster)

Primary Key: event_ID, client_ID
Foreign Key: event_ID References events(event_ID)
Foreign Key: client_ID References clients(client_ID)

events_has_venues(event_ID, venue_ID, start_date, end_date)

Primary Key: event_ID, venue_ID
Foreign Key: event_ID References events(event_ID)
Foreign Key: venue_ID References venues(venue_ID)

SQL QUERIES EXAMPLES

Query 1:

Ticket Price Distribution and Category Analysis:

This query analyzes the distribution of ticket prices across event categories, providing insights into pricing strategies for different types of events.

```
SELECT
    e.event_category,
    COUNT(t.ticket_ID) AS num_tickets_sold,
    MIN(t.ticket_price) AS min_ticket_price,
    MAX(t.ticket_price) AS max_ticket_price,
    ROUND(AVG(t.ticket_price), 2) AS avg_ticket_price
FROM
    events e
JOIN
    tickets t ON e.event_ID = t.event_ID
GROUP BY
    e.event_category
ORDER BY
    num_tickets_sold DESC;
```

Event Category	Number of Tickets Sold	Min Ticket Price	Max Ticket Price	Avg Ticket Price
Workshop	16	45	100	69.06
Festival	7	45	75	56.43
Exhibition	6	50	100	74.17
Fair	6	55	100	75.00
Performance	6	45	95	73.33

SQL QUERIES EXAMPLES

Query 2:

Revenue Distribution Between Clients and Ticketmaster:

This query calculates the revenue distribution by percentage between clients and Ticketmaster for each event, considering the payments made to clients and Ticketmaster for each event.

Event Name	Event Category	Clients Revenue	Business Revenue	Clients Percentage	Business Percentage
Robotics Competition	Competition	1280.18	256.03	83.333659	16.666341
Fitness Bootcamp	Bootcamp	1025.47	205.09	83.333604	16.666396
Gastronomy Event	Event	1379.62	275.92	83.333535	16.666465
Fashion Design Workshop	Workshop	1424.72	284.94	83.333528	16.666472
Comedy Night	Performance	1730.37	346.07	83.333494	16.666506
Outdoor Painting	Art	1891.17	378.23	83.333480	16.666520

```
SELECT
    e.event_name,
    e.event_category,
    SUM(ec.payment_to_client) AS revenue_to_clients,
    SUM(ec.payment_to_ticketmaster) AS
revenue_to_ticketmaster,
    (SUM(ec.payment_to_client) /
(SUM(ec.payment_to_client) +
SUM(ec.payment_to_ticketmaster))) * 100 AS
client_percentage,
            (SUM(ec.payment_to_ticketmaster) /
(SUM(ec.payment_to_client) +
SUM(ec.payment_to_ticketmaster))) * 100 AS
ticketmaster_percentage

FROM
    events e
JOIN
    events_has_clients ec ON e.event_ID = ec.event_ID
GROUP BY
    e.event_ID
ORDER BY
    client_percentage DESC;
```

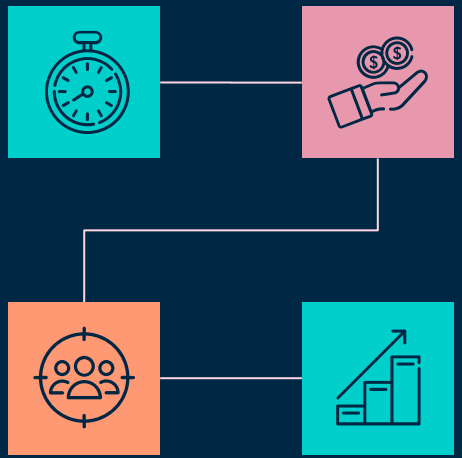
LEARNING EXPERIENCE: ISSUES & CHALLENGES

TIME LIMIT

We have a very limited work time for the whole project

TEAMWORK

We have different working schedules and may be hard to accommodate each others



NEW SKILLS

We have to apply new skills such as building a webpage

MEANINGFUL DEVELOPMENT

How we can build insights on these data and make useful queries that can further help the business development

LEARNING EXPERIENCE: SOLUTIONS

Plan out the work and timeline at the beginning and set norms to make sure members can get work done on time

TIME

TEAMWORK

Let members who have prior knowledge on building website start to work on the part to save time learning

SKILLS

INSIGHTS

Frequently communicate in group chat to keep track on our project and also set up touchpoint to ensure timeline and members feel comfortable doing tasks

We consider multiple situations and select the most meaningful queries to make sure we can make use of these data to help the business

EXTENDING OUR WORK

Security Enhancements and Backup

Ensure secured data storage; implementing more rigorous user authentication systems; and securing ticket transfer or resale methods

Implement Data Archiving and Purging

Identify historical data that is no longer frequently accessed but needs to be retained for compliance or reference purposes. Move such data to an archival database or storage while keeping the main operational database optimized for current transactions.

Scalability and Performance Optimization

Consider implementing database scaling solutions to accommodate future growth in data and user traffic. Implement techniques such as sharding, partitioning, and caching to optimize database performance and reduce latency

THANKS

Do you have any questions?