Contents

1. Business Context and Database Design 1
1.1 Business Context Definition 1
1.2 The Data Product 2
1.3 Expected Reports & Business Insights 2
2. Database Design 3
2.1 Entity-Relationship Diagram (ERD) 3
2.2 ERD Design Considerations & Rationale 4
2.3 Schema Design 4
2.4 Design Assumptions 5
2.5 Table Relationship Cardinality 6
3. SQL Schema Implementation 6
4. Synthetic Data Generation 7
5. Business Insights & Report Generation 8
6. Limitations & Recommendations 11
7. Appendix 12

1. Business Context and Database Design

1.1 Business Context Definition

1.1.1 Company Overview

OneGym is a young gym chain operating across seven West Midlands, UK branches. Each branch offers a range of standard gym memberships and access to nine fitness classes. Branches are staffed by two trainers minimum, with additional staff based on branch size. Each trainer is responsible for general duties and conducting a specific class.

1.1.2 Key Business Functions & Processes

- **Membership Registration & Management:** Members join a specific branch and purchase a membership plan. The system records core member information which enables analysis of sign-up trends, renewals, and membership distribution.

- **Trainer Management & Class Delivery:** Trainers are assigned to one branch and have a single specialization (i.e. Strength, Cardio), leading scheduled group class sessions with relevant associated details.

- **Class Attendance & Feedback:** Members can attend class sessions and optionally provide a 1-5 rating, supporting analysis of class popularity, capacity utilization, and satisfaction.

- **Check-Ins & Visit Ratings:** Members check in and out of the gym, creating timestamped logs. After visits, they can rate their experience, providing branch-level satisfaction data.

- **Payment Processing:** Each membership purchase creates a payment record, supporting revenue analysis, purchase trends, and customer loyalty tracking.

1.2 The Data Product

The primary purpose of our data product is to facilitate decision-making by enabling the generation of business-critical insights from gym operations. Tracking key performance metrics across branch, memberships, understanding customer behaviour, and evaluating the effectiveness of gym services, consequently, support the following:

- **Operational visibility:** Tracking member interaction with gym services through check-ins, class attendance, and feedback ratings.

- **Revenue & financial analysis:** Revenue derived from memberships across membership types and branches.

- **Staff performance & resource allocation:** Trainer workloads, session ratings, and optimizing demand-based trainer allocation.

- **Customer engagement & retention:** Membership renewal and churn patterns potentially correlated to customer satisfaction and engagement.

- **Strategic planning:** Potential future implementation of promotions, improved staffing schedules, and branch-specific development.

1.3 Expected Reports & Business Insights

The database and resultant data product enable the extraction of various key data-driven business insights, examples include but not exhausted to the following:

- **Membership and Retention Reports:**
    - Membership trends by type and branch
    - Renewal and churn patterns
    - Cumulative membership-derived revenue

- **Financial Performance Reports:**
    - Revenue by membership type
    - Revenue by branch

- **Facility and Usage Analytics:**
    - Seasonal check-in patterns
    - Peak and off-peak patterns
    - Visit frequency and member engagement

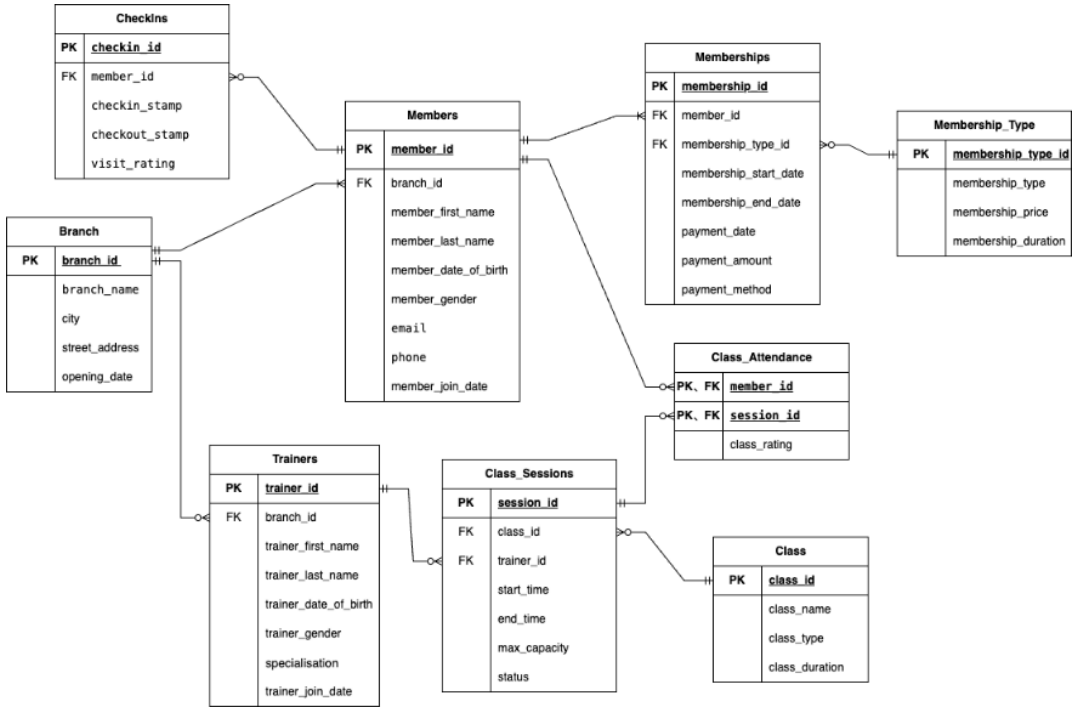
- **Class and Trainer Insights:**
    - Class popularity and utilization
    - Session attendance and feedback
    - Trainer utilization

- **Member Satisfaction Reports:**
    - Branch visit ratings
    - Class ratings
    - Identification of underperformance

2. Database Design

2.1 Entity-Relationship Diagram (ERD)

The ERD illustrates the core structure of OneGym's database system, capturing key entities, relationships, and business logic of a multi-branch gym chain. At the centre of the model is the **Members** table, representing the customer base and linking directly or indirectly to all other operational tables.



- **Branch**: Represents individual gym locations.
- **Memberships**: Captures individual membership purchases, payment info, and duration.
- **Membership_Type**: Defines the structure (Monthly, Quarterly, Annual) and pricing of plans.
- **CheckIns**: Tracks member gym visits and optional visit ratings.
- **Trainers**: Staff assigned to branches and leading class sessions.
- **Class_Sessions**: Class occurrences (including trainer, time, capacity).
- **Class**: Detailing the class name and its corresponding class type.
- **Class_Attendance**: Junction table linking members to class sessions, with optional ratings.

2.2 ERD Design Considerations & Rationale

In designing the ERD, several key considerations shaped our approach:

- **Realistic Scope Representation (Mini-World)**: The model mirrors aspects of real gym-chain operations. All entities reflect typical gym entities, enabling meaningful insight generation and practical decision-making.

- **Balancing Complexity & Practicality**: The model is sufficiently rich in relationships (i.e. M:N and 1:N relationships) while avoiding unnecessary complexity.

- **Referential Integrity & Normalisation**: The ERD ensures referential integrity via foreign key relationships and adherence to Third Normal Form (3NF). The design avoids redundancy while retaining practicality.

- **Designing for Business Value**: The ERD provides clear understanding of the business use case. Each relationship and attribute are utilised for downstream reporting and decision-making to feed into business KPIs.

2.3 Schema Design

The schema represents the logical structure and implementation of OneGym's database, translating the ERD into a compilation of tables, fields, data types, primary keys, and foreign key relationships. The schema builds on the ERD by detailing data storage, use and validation forming the foundation for efficient querying and analysis. Below is the example Branch table schema; the full schema located in the appendix (Appendix 1).

Branch: *Basic information about branches.*

Attribute	Data Type	Key/Constraints	Description
branch_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the branch.
branch_name	TEXT	NOT NULL, UNIQUE	Branch's name; Should be unique.
city	TEXT	NOT NULL	City the branch is in.
street_address	TEXT	NOT NULL, UNIQUE	Detailed street address; Should be unique.
opening_date	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD)	Opening date in standardized format.

2.4 Design Assumptions

In addition to the database schema and to ensure data integrity and logical consistency for synthetic data generation, a comprehensive list of assumptions was created. It reflects both system-level constraints and table-specific rules.

2.4.1 General Assumptions (Database-Level)

- The gym operates multiple branches, and each member and trainer can be assigned to exactly one branch.
- A trainer must belong to a branch; they cannot exist independently.
- A trainer can lead multiple sessions, but each session is led by only one trainer.
- A member must have an active membership to check in at the gym.
- A member's first payment signifies their join date. Later memberships do not trigger new account IDs.
- Trainers do not hold memberships – they are staff, not customers.
- A class is a general offering (i.e., "Yoga"), while a class session represents a scheduled occurrence of that class.
- Each trainer has only one specialization and can only teach those classes (i.e., Cardio, Strength).
- All data entries (i.e., memberships, sessions, check-ins) must occur after the opening date of the assigned branch, except for date of birth values.
- OneGym's only source of income right now is memberships.

Table-level assumptions can be found in the appendix (Appendix 2).

2.5 Table Relationship Cardinality

The cardinalities accurately reflect real-world operations of a multi-branch gym chain, ensuring realistic and scalable data modelling, while preserving simplicity and referential integrity:

Those acquiring a One to Many (1:M) relationship:

Branch – Members

Branch – Trainer

Members – Membership

Membership Type – Membership

Members – CheckIns

Trainer – Class Sessions

Class – Class Sessions

Those acquiring a Many to Many (M:M) relationship:

Members – Class Sessions (via Class Attendance)

3. SQL Schema Implementation

The primary goal of our database schema was to enforce data consistency, eliminate redundancy, and ensure referential integrity, adopting appropriate use of primary keys, foreign keys, constraints, and normalisation principles.

The schema complies with at least Third Normal Form (3NF); thus, each table contains only atomic attributes, no partial or transitive dependencies, and clear relationships are maintained through foreign keys. Each entity has been assigned an appropriate SQLite data type based on the nature of the data, keeping the design efficient and consistent.

The schema enforces data integrity, normalization (1NF–3NF), and business rules through various constraints. INTEGER is used for auto-incremented primary keys, ensuring uniqueness. TEXT is applied to fields like names, addresses, and timestamps, with CHECK constraints enforcing correct formats (e.g., ISO 8601 for dates and emails containing "@").

NOT NULL constraints ensure all business-critical data is mandatory, while UNIQUE constraints prevent duplication (e.g., branch names, street addresses, membership types). FOREIGN KEYS establish relationships, such as 1:N links between trainers and branches or members and memberships, maintaining referential integrity.

Special constraints refine data accuracy, such as CHECK on class duration (30–60 mins), membership types (annual, quarterly, monthly), and visit/session ratings (1–5). REAL data types handle pricing and payments, with CHECK constraints preventing negative values.

Composite PRIMARY KEYS in attendance tracking avoid duplicate records, and CHECK ensures logical constraints (e.g., end_time > start_time). The schema remains scalable and structured, supporting future expansion while avoiding redundancy.

Some of these implementation methods can be exemplified in the image below associated with the Class Sessions Table from the dataset.

```
-- Create Class Sessions table
CREATE TABLE Class_Sessions (
    session_id INTEGER PRIMARY KEY AUTOINCREMENT,
    class_id INTEGER NOT NULL,
    trainer_id INTEGER NOT NULL,
    start_time TEXT NOT NULL CHECK(start_time LIKE '____-__-__ __:__:__'),
    end_time TEXT NOT NULL CHECK(end_time > start_time),
    max_capacity INTEGER CHECK(max_capacity > 0),
    status TEXT NOT NULL CHECK(status IN ('Completed', 'Cancelled')),
    FOREIGN KEY (class_id) REFERENCES Class(class_id),
    FOREIGN KEY (trainer_id) REFERENCES Trainers(trainer_id)
);
```

Examples of each table that was created during the data generation process is found in the appendix (Appendix 3).

4. Synthetic Data Generation

The synthetic data for this project was generated using the Faker library in Python. The goal was not only to populate the database, but to ensure that the data could simulate meaningful business scenarios and enable insights for effective decision-making across key business areas.

The data generation process was highly iterative. Initial versions focused on aligning with the designed schema and assumptions, ensuring referential integrity across tables. However, as the project progressed, additional business rules and logical constraints were created (i.e., check-in timestamps must fall within membership periods, or a trainer cannot be double-booked). Each iteration improved data quality and completeness.

One key challenge was balancing technical validity with business usefulness. Even after generating working data, manual adjustments were often required to introduce “interesting” business patterns – such as class ratings varying across branches, or fluctuating visit activity across seasons. These enhancements made the database more insightful but also required constant cross-checking to ensure all assumptions and dependencies still held.

Ultimately, this combination of logical structure and analytical richness ensures the data product is both technically robust and business relevant. Example rows for each table can be found in the appendix. The code for data generation is attached to the assignment.

5. Business Insights & Report Generation

This section outlines a series of SQL queries developed to generate business insights aligned with OneGym’s operational and strategic objectives. All insights are created from the period between 2022-2024. The purpose of each and consequent insights are detailed throughout. Corresponding visualisations are included in the appendix (Appendix 5).

1. Total Membership Numbers by Branch

```sql
SELECT
  b.branch_name,
  strftime('%Y', ms.payment_date) AS payment_year,
  COUNT(ms.payment_date) AS total_memberships_sold
FROM Memberships ms
JOIN Members m ON ms.member_id = m.member_id
JOIN Branch b ON m.branch_id = b.branch_id
WHERE strftime('%Y', ms.payment_date) IN ('2022', '2023', '2024')
GROUP BY b.branch_name, payment_year
ORDER BY b.branch_name, payment_year;
```

Tracking overall membership volume across branches over time – key for evaluating gym performance and engagement. Memberships are declining consistently across all branches. This signals an urgent need for intervention – marketing campaigns, promotions, or improving gym offerings.

2. Memberships by Type

```
SELECT
  strftime('%Y', ms.payment_date) AS payment_year,
  mt.membership_type,
  SUM(ms.payment_amount) AS total_membership_revenue
FROM Memberships ms
JOIN Membership_Type mt ON ms.membership_type_id = mt.membership_type_id
    WHERE strftime('%Y', ms.payment_date) IN ('2022', '2023', '2024')
GROUP BY payment_year, mt.membership_type
ORDER BY payment_year, mt.membership_type;
```

Evaluating whether the decline in memberships is type-specific or systemic. All types show similar downward trends, suggesting the issue is structural rather than related to specific plans. Urgent review of overall value proposition is required.

3. Revenue by Membership Type

```
SELECT
  strftime('%Y', ms.payment_date) AS payment_year,
  mt.membership_type,
  SUM(ms.payment_amount) AS total_membership_revenue
FROM Memberships ms
JOIN Membership_Type mt ON ms.membership_type_id = mt.membership_type_id
    WHERE strftime('%Y', ms.payment_date) IN ('2022', '2023', '2024')
GROUP BY payment_year, mt.membership_type
ORDER BY payment_year, mt.membership_type;
```

Assessing membership-type contributions to total revenue.

Revenue is heavily driven by annual plans. Focused efforts are needed to retain and grow this high-value segment, while also exploring ways to promote shorter term memberships.

4. Class Popularity by Branch

```
SELECT
  b.branch_name,
  c.class_type,
  COUNT(ca.member_id) AS total_attendance
FROM Class_Attendance ca
JOIN Class_Sessions cs ON ca.session_id = cs.session_id
JOIN Class c ON cs.class_id = c.class_id
JOIN Trainers t ON cs.trainer_id = t.trainer_id
JOIN Branch b ON t.branch_id = b.branch_id
WHERE strftime('%Y', cs.start_time) IN ('2022', '2023', '2024') AND cs.status = 'Completed'
GROUP BY b.branch_name, c.class_type
ORDER BY b.branch_name, total_attendance DESC;
```

Identifying class type preferences across branches to optimize scheduling and promotion.

Flexibility classes dominate attendance, especially at Birmingham. Insights can help scale popular classes and/or promote underperforming ones.

5. Gym Attendance Patterns across Seasons

```sql
SELECT
  b.branch_name,
  strftime('%Y', c.checkin_stamp) AS year,
  strftime('%m', c.checkin_stamp) AS month,
  COUNT(c.checkin_id) AS total_checkins
FROM CheckIns c
JOIN Members m ON c.member_id = m.member_id
JOIN Branch b ON m.branch_id = b.branch_id
WHERE strftime('%Y', c.checkin_stamp) IN ('2022', '2023', '2024')
GROUP BY b.branch_name, year, month
ORDER BY b.branch_name, year, month;
```

Understanding seasonal gym usage trends for resource allocation.

Winter months show peak activity, partly due to yearly wave of “new year’s resolution” peaks. Branches should adjust class schedules, staffing, and marketing efforts accordingly.

6. Churn Trends by Branch

```sql
SELECT
  b.branch_name,
  strftime('%Y', ms.membership_end_date) AS membership_end_year,
  COUNT(DISTINCT m.member_id) AS churned_members
FROM Members m
JOIN Branch b ON m.branch_id = b.branch_id
JOIN Memberships ms ON m.member_id = ms.member_id
JOIN (
  SELECT
  member_id,
  MAX(strftime('%Y', membership_end_date)) AS max_end_year
    FROM Memberships
    GROUP BY member_id
  ) AS max_dates ON m.member_id = max_dates.member_id
WHERE strftime('%Y', ms.membership_end_date) IN ('2022', '2023', '2024')
AND strftime('%Y', ms.membership_end_date) = max_dates.max_end_year
GROUP BY membership_end_year, b.branch_name
ORDER BY b.branch_name, membership_end_year;
```

Tracking member retention performance at branch level.

Churn rates appear to decrease, but this may be misleading due to falling total memberships. Deeper analysis of member behaviour is required.

Further metrics are provided in the appendix (Appendix 4) for reference.

This insights and their related SQL queries can drive meaningful business insights and operational improvements. The reports align directly with OneGym’s core objectives – from monitoring membership and revenue trends to enhancing customer satisfaction and optimizing class delivery. These insights not only enable data-informed decisions but also help shape future strategy across branches.

6. Limitations & Recommendations

Despite offering a solid foundation for analysis, the current database design reflects a simplified "mini world" and comes with multiple limitations:

Limitations

- **Simplified Scope**: The model abstracts real-life gym operations. Many more complex processes like membership pauses, promotions, or staff schedules were out of scope and not included.

- **Synthetic Data Constraints**: The dataset was artificially created, requiring manual adjustments to simulate meaningful trends while maintaining referential integrity.

- **Limited Entity Relationships**: All relationships and business logic are tightly defined and may not fully capture real-world scenarios (i.e. one branch per member, one class type per trainer).

- **No Historical Tracking**: Changes over time (i.e. price updates) are not tracked, limiting historic and long-term analysis.

- **Assumption Dependency**: The system depends on fixed assumptions that may restrict scalability or adaptability to evolving and changing requirements.

Recommendations

- **Expand Gradually**: Introduce new entities and attributes iteratively to enhance realism i.e. promotions or paid private sessions with trainers.

- **Track Historical Data**: Add mechanisms for tracking changes over time, such as price history or trainer roles.
- **Increase Relationship Flexibility**: Allow for multi-branch access, multiple class specializations, or more complex class booking systems.

- **Future Adaptability**: Align schema development as data and business needs evolve and change.

7. Appendix

1. Database Schema

Branch: *Basic information about branches.*

Attribute	Data Type	Key/Constraints	Description
branch_id	INTEGER	PRIMARY KEY,AUTOINCREMENT	Unique identifier for the branch.
branch_name	TEXT	NOT NULL, UNIQUE	Branch's name; Should be unique.
city	TEXT	NOT NULL	City the branch is in.
street_address	TEXT	NOT NULL,UNIQUE	Detailed street address; Should be unique.
opening_date	TEXT	NOT NULL,CHECK(ISO8601 format: YYYY-MM-DD)	Opening date in standardized format.

Members: *Stores information about gym members, including their personal details and associated branch/trainer.*

Attribute	Data Type	Keys/Constraints	Description
member_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the member.
branch_id	INTEGER	FOREIGN KEY REFERENCES Branch(branch_id), NOT NULL	Associates the member with a specific branch.
member_first_name	TEXT	NOT NULL	Member's first name.
member_last_name	TEXT	NOT NULL	Member's last name.
member_date_of_birth	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD)	Member's birthdate in standardized format.
member_gender	TEXT	NOT NULL, CHECK(gender IN ('M', 'F', 'Other'))	Member's gender assignment.
email	TEXT	NOT NULL, CHECK(email LIKE '%@%.%'), UNIQUE	Valid email format. Unique Identifier, must not be duplicated between members.
phone	TEXT	NOT NULL, CHECK(LENGTH(phone) = 10), UNIQUE	Phone number with 10 digits. Unique Identifier, must not be duplicated between members.

member_join_date	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD), CHECK (DATE_OF_BIRTH <= DATE_SUB(CURDATE(), INTERVAL 18 YEAR))	The date of the first ever payment transaction (payment_date in Memberships) made by the corresponding member_id.

Trainers: *Stores information about gym trainers, including their personal details, specialisations, and associated branch.*

Attribute	Data Type	Keys/Constraints	Description
trainer_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the trainer.
branch_id	INTEGER	FOREIGN KEY REFERENCES Branch(branch_id), NOT NULL	Associates the trainer with a specific branch.
trainer_first_name	TEXT	NOT NULL	Trainer's first name.
trainer_last_name	TEXT	NOT NULL	Trainer's last name.
trainer_gender	TEXT	NOT NULL, CHECK(gender IN ('M', 'F', 'Other'))	Trainer's gender assignment.
trainer_date_of_birth	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD)	Trainer's birthdate in standardized format.
specialisation	TEXT	NOT NULL	Trainer's area of expertise.
trainer_join_date	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD)	Date the trainer joined the gym.

Class: *Stores fitness classes offered by the gym, including their type and duration.*

Attribute	Data Type	Keys/Constraints	Description
class_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the class.
class_name	TEXT	NOT NULL, UNIQUE	Class name (e.g., "Core").
class_type	TEXT	NOT NULL,	Category of the class (enforced values).
class_ duration	INTEGER	NOT NULL, CHECK(duration BETWEEN 30 and 60)	Duration in minutes.

Membership Type:*Stores fixed-duration membership plans with predefined pricing (monthly, quarterly, annual).*

Attribute	Data Type	Keys/Constraints	Description
membership_type_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the membership plan.
membership_type	TEXT	NOT NULL, UNIQUE	Membership tier (enforced values).
membership_price	REAL	NOT NULL	Predefined price for each tier.
membership_duration	INTEGER	NOT NULL	Predefined period in months for each tier.

Memberships: *Records financial transactions related to memberships.*

Attribute	Data Type	Keys/Constraints	Description
membership_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the membership payment transaction.
member_id	INTEGER	FOREIGN KEY REFERENCES Members(member_id), NOT NULL	Links to the member associated with the payment.
membership_type_id	INTEGER	FOREIGN KEY REFERENCES Membership Type(membership__type_id), NOT NULL	Must be linked to a member_id. Membership_type_id must be provided.
membership_start_date	TEXT	NOT NULL,CHECK(ISO8601 format: YYYY-MM-DD)	Membership start date in standardized format.
membership_end_date	TEXT	NOT NULL,CHECK(ISO8601 format: YYYY-MM-DD)	Membership end date in standardized format. (Membership start date + membership_duration)
payment_date	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD)	Date of the payment transaction.
payment_amount	REAL	NOT NULL, CHECK(payment_amount >= 0)	Transaction amount. Must match the 'price' found in Membership_Type
payment_method	TEXT	NOT NULL	Payment method.

Check-Ins: *Records gym member check-ins and check-outs, including timestamps and optional feedback ratings.*

Attribute	Data Type	Keys/Constraints	Description
checkin_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the check-in record.
member_id	INTEGER	FOREIGN KEY REFERENCES Members(member_id), NOT NULL	Links to the member who checked in.
checkin_stamp	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD HH:MM:SS)	Timestamp when the member entered the gym.
checkout_stamp	TEXT	CHECK(ISO8601 format: YYYY-MM-DD HH:MM:SS)	Timestamp when the member exited the gym.
visit_rating	INTEGER	CHECK(rating BETWEEN 1 AND 5)	Optional member rating for the visit (1-5 stars).

Class Sessions: *Records scheduled group fitness class sessions, including trainer assignments and session status.*

Attribute	Data Type	Keys/Constraints	Description
session_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the class session.
class_id	INTEGER	FOREIGN KEY REFERENCES Class(class_id), NOT NULL	Links to the Class.
trainer_id	INTEGER	FOREIGN KEY REFERENCES Trainer(trainer_id), NOT NULL	Links to the assigned trainer.
start_time	TEXT	NOT NULL, CHECK(ISO8601 format: YYYY-MM-DD HH:MM:SS)	Scheduled start time of the class session.
end_time	TEXT	NOT NULL, CHECK(end_time > start_time)	Scheduled end time of the class session.
max_capacity	INTEGER	CHECK(max_capacity >0)	The maximum capacity of attendees for each class.
Status	TEXT	NOT NULL	The status of the class, can be either 'Completed' or 'Cancelled'.

Class Attendance*: Records member attendance and feedback for scheduled class sessions.*

Attribute	Data Type	Keys/Constraints	Description

member_id	INTEGER	PRIMARY KEY, FOREIGN KEY REFERENCES Members(member_id)	Links to the attending member.
session_id	INTEGER	PRIMARY KEY, FOREIGN KEY REFERENCES Class_Sessions(session_id)	Links to the scheduled class session.
class_rating	INTEGER	CHECK(rating BETWEEN 1 AND 5)	Member’s feedback rating for each class session attended (1-5 stars, optional)

Class: *Stores fitness classes offered by the gym, including their type and duration.*

Attribute	Data Type	Keys/Constraints	Description
class_id	INTEGER	PRIMARY KEY, AUTOINCREMENT	Unique identifier for the class.
class_name	TEXT	NOT NULL, UNIQUE	Class name (e.g., "Core").
class_type	TEXT	NOT NULL,	Category of the class (enforced values).
class_ duration	INTEGER	NOT NULL, CHECK(duration BETWEEN 30 and 60)	Duration in minutes.

2. Table-Level Assumptions

1. Branch Table

- Each branch must be located in the West Midlands.
- All branch names must include the city they are in.
- A branch cannot be deleted if trainers or members are still assigned.
- A member's or trainer's join/start date must be after the branch’s opening date.
- A member can only be assigned to one branch. If they switched branches, they would be considered a new member with a new ID at that branch. There are no branch transfers.

2. Members Table

- Every member must belong to exactly one branch.
- All personal details (name, DOB, gender, email, phone) are mandatory.
- The join date is always equal to first membership payment date.
- A member is considered active if membership_end_date is in the future.
- A member must be at least 18 years old.

3. Trainers Table

- Each trainer belongs to exactly one branch.

- Each trainer has exactly one specialization, chosen from a predefined list.
- The join date must be after the branch opening date.
- Trainer age must be at least 18.

4. Class Table

- Each class represents a class type, not a session.
- Class duration must be between 30 and 60 minutes in increments of 15 minutes.
- The duration of a class session must match the class duration exactly.

5. Memberships Table

- Each membership type has a fixed price and duration. There are no discounts.
- A member can only have one active membership at a time.
- A new membership can be purchased before the current one ends. In case of a payment before the current membership has ended, the new membership does not start until the end of the previous one plus one day.

7. Check_Ins Table

- Each check-in/check-out must link to a valid member_id.
- Check-ins can only occur during active memberships.
- Checkout_stamp must always be after checkin_stamp.
- Optional visit_rating can only be entered after checkout.

8. Class_Sessions Table

- Each session must be linked to a class and a trainer.
- Check-in time must be before class session start_time and check-out time must be after class session end_time.
- A trainer cannot be double-booked (no overlapping sessions).
- Class attendance must always be smaller or equal to max capacity.
- If a class was booked by a member, it is assumed "Completed" unless marked as "Cancelled" by the gym. “Cancelled” sessions have no attendance.

3. Extracts of created Data Tables

Branch

A	B	C	D	E
branch_id	branch_name	city	street_address	opening_date
1	Birmingham-High Street	Birmingham	123 High Street	1/10/2020
2	Coventry-Broadgate	Coventry	45 Broadgate	4/18/2020
3	Wolverhampton-Queen Square	Wolverhampton	78 Queen Square	6/12/2020
4	Dudley-Church Street	Dudley	56 Church Street	9/5/2020
5	Solihull-Touchwood Road	Solihull	89 Touchwood Road	3/22/2021
6	Walsall-Park Street	Walsall	34 Park Street	7/25/2021
7	West Bromwich-New Square	West Bromwich	12 New Square	11/14/2021

CheckIns

A	B	C	D	E
checkin_id	member_id	checkin_stamp	checkout_stamp	visit_rating
1	1	11/16/2021 19:00	11/16/2021 21:01	
2	1	11/16/2021 17:15	11/16/2021 18:33	
3	1	11/19/2021 19:00	11/19/2021 19:44	
4	1	11/21/2021 7:45	11/21/2021 9:28	
5	1	11/26/2021 19:00	11/26/2021 21:05	
6	1	11/29/2021 11:45	11/29/2021 14:21	
7	1	12/2/2021 9:00	12/2/2021 11:02	
8	1	12/4/2021 18:15	12/4/2021 20:34	
9	1	12/9/2021 20:30	12/9/2021 21:35	5

Class

A	B	C	D
class_id	class_name	class_type	class_duration
1	Boxercise	Cardio	30
2	Yoga Flow	Flexibility	45
3	Full Body Stretch	Stretching	45
4	Advanced Flex	Flexibility	45
5	Core Crusher	Strength	30
6	Deep Stretching	Stretching	45
7	Endurance Ride	Cardio	60
8	Power Lifting	Strength	45
9	Flexibility Basics	Flexibility	60

Class_Attendance

A	B	C
member_id	session_id	class_rating
17	5	4
52	5	3
336	5	
17	6	1
298	6	1
286	7	4
201	10	2
107	12	
121	12	2

Class_Sessions

A	B	C	D	E	F	G
session_id	class_id	trainer_id	start_time	end_time	max_capacity	status
1	2	3	1/1/2022 18:30	1/1/2022 19:15	20	Cancelled
2	3	2	1/1/2022 9:00	1/1/2022 9:45	10	Completed
3	5	1	1/1/2022 7:00	1/1/2022 7:30	10	Completed
4	3	1	1/2/2022 7:00	1/2/2022 7:45	15	Completed
5	5	2	1/2/2022 19:00	1/2/2022 19:30	10	Completed
6	6	3	1/2/2022 9:00	1/2/2022 9:45	20	Completed
7	2	2	1/3/2022 19:00	1/3/2022 19:45	15	Completed
8	6	3	1/3/2022 9:00	1/3/2022 9:45	15	Cancelled
9	3	1	1/3/2022 7:30	1/3/2022 8:15	10	Completed

Members

A	B	C	D	E	F	G	H	I
member_id	branch_id	member_first_name	member_last_name	member_date_of_birth	member_gender	email	phone	member_join_date
1	1	Priscilla	Walsh	1/4/1956	F	x932afs3adn0@yahoo.com	7930330818	11/13/2021
2	1	Stephanie	Bender	6/19/1954	F	cgd3k0gp3q8h@icloud.com	7857116626	4/27/2021
3	1	Leonard	Solis	11/15/1952	M	ioe2np5v@gmail.com	7129065437	6/26/2020
4	1	Lisa	Owen	1/11/1953	F	7qt2fpgfn5e1@hotmail.com	7480159414	6/6/2020
5	1	William	Richards	6/2/1947	M	ajq425@gmail.com	7807147361	3/7/2022
6	1	Gavin	Gomez	6/17/1965	M	k2zzf66eawx8@hotmail.com	7450014558	12/31/2024
7	1	Wesley	Ellis	6/15/1945	M	eman0676p@hotmail.com	7805599199	9/20/2021
8	1	Casey	Clark	1/25/1958	M	3yk590vvu@gmail.com	7345325843	5/10/2023
9	1	Tanner	Oconnor	8/15/1964	M	q86r37l32t@gmail.com	7992384347	7/23/2020
10	1	Benjamin	Macias	10/16/1949	M	9sreq12v2@icloud.com	7753572689	1/21/2022

Membership_Type

A	B	C	D
membership_type_id	membership_type	membership_price	membership_duration
1	Monthly	20	30
2	Quarterly	50	90
3	Annual	180	365

Memberships

A	B	C	D	E	F	G	H
membership_id	member_id	membership_type_id	membership_start_date	membership_end_date	payment_date	payment_amount	payment_method
1	1	2	11/13/2021	2/11/2022	11/13/2021	50	Credit Card
2	2	2	4/27/2021	7/26/2021	4/27/2021	50	Paypal
3	2	3	10/29/2021	10/29/2022	10/29/2021	180	Paypal
4	3	3	6/26/2020	6/26/2021	6/26/2020	180	Cash
5	3	3	7/15/2021	7/15/2022	7/15/2021	180	Debit Card
6	4	2	6/6/2020	9/4/2020	6/6/2020	50	Credit Card
7	4	1	1/3/2021	2/2/2021	1/3/2021	20	Debit Card
8	4	1	2/3/2021	3/5/2021	1/30/2021	20	Paypal
9	4	1	3/6/2021	4/5/2021	2/28/2021	20	Credit Card

Trainers

A	B	C	D	E	F	G	H
trainer_id	branch_id	trainer_first_name	trainer_last_name	trainer_gender	trainer_date_of_birth	specialisation	trainer_join_date
1	1	Devin	Gilmore	M	11/19/1997	Posture	5/26/2021
2	1	Jacqueline	Horton	F	1/15/1983	Rehabilitation	2/4/2020
3	1	Cheyenne	Jones	F	6/14/1993	Cardio	3/19/2021
4	2	John	Mullins	M	8/21/1993	Strength	5/30/2020
5	2	Thomas	Lee	M	2/4/1987	Rehabilitation	1/6/2021
6	2	Jessica	Salazar	F	7/5/1997	Posture	7/5/2020
7	2	Bryan	Kim	M	6/29/1981	Cardio	9/7/2020
8	3	Jessica	Peters	F	3/28/1998	Strength	12/26/2021
9	3	Alexander	Daniels	M	8/31/1980	Cardio	10/10/2021
10	4	Debra	Goodwin	F	7/23/1990	Cardio	11/14/2020

4. Additional Metrics & Insights

7. Churn Trends by Membership Type

```
SELECT
  b.membership_type,
  strftime('%Y', ms.membership_end_date) AS membership_end_year,
  COUNT(DISTINCT m.member_id) AS churned_members
FROM Members m
JOIN Memberships ms ON m.member_id = ms.member_id
JOIN Membership_Type b ON ms.membership_type_id = b.membership_type_id
JOIN (
  SELECT
  member_id,
  MAX(strftime('%Y', membership_end_date)) AS max_end_year
  FROM Memberships
  GROUP BY member_id
  ) AS max_dates ON m.member_id = max_dates.member_id
WHERE  strftime('%Y', ms.membership_end_date) IN ('2022', '2023', '2024')
AND strftime('%Y', ms.membership_end_date) = max_dates.max_end_year
GROUP BY membership_end_year, b.membership_type
ORDER BY b.membership_type, membership_end_year;
```

Purpose:

Measures churn levels across different membership types.

Insights & Actionability:

Annual memberships show rising churn. May indicate preference shift toward shifting preferences to short-term flexibility – further qualitative research recommended.

8. Visit Rating Split by Branch

```sql
WITH CheckinCounts AS (
    SELECT
        b.branch_name,
        ci.visit_rating,
        COUNT(ci.checkin_id) AS rating_count
    FROM CheckIns ci
    JOIN Members m ON ci.member_id = m.member_id
    JOIN Branch b ON m.branch_id = b.branch_id
    WHERE strftime('%Y', ci.checkin_stamp) IN ('2022', '2023', '2024')
      AND ci.visit_rating IS NOT NULL
    GROUP BY b.branch_name, ci.visit_rating
),
BranchTotal AS (
    SELECT
        branch_name,
        count(*) AS total_checkins
    FROM CheckIns ci
    JOIN Members m ON ci.member_id = m.member_id
    JOIN Branch b ON m.branch_id = b.branch_id
    WHERE strftime('%Y', ci.checkin_stamp) IN ('2022', '2023', '2024')
      AND ci.visit_rating IS NOT NULL
    GROUP BY branch_name
)
SELECT
    c.branch_name,
    c.visit_rating,
    c.rating_count * 1.0 / b.total_checkins *100 AS rating_percentage
FROM CheckinCounts c
JOIN BranchTotal b ON c.branch_name = b.branch_name
ORDER BY c.branch_name, c.visit_rating;
```

Purpose:

Represents customer satisfaction per branch using post-visit ratings.

Insights & Actionability:

Dudley-Church Street leads with 5-star ratings, suggesting it excels in best practices worth replicating across other branches. Branches with low ratings need intervention.

9. Average Visit Ratings by Branch

```sql
SELECT
  b.branch_name AS branch_name,
  ROUND(AVG(c.visit_rating), 3) AS avg_visit_rating
FROM CheckIns c
JOIN Members m ON c.member_id = m.member_id
JOIN Branch b ON m.branch_id = b.branch_id
WHERE strftime('%Y', c.checkin_stamp) IN ('2022', '2023', '2024')
  AND c.visit_rating IS NOT NULL
GROUP BY b.branch_name
ORDER BY avg_visit_rating DESC;
```

Purpose:

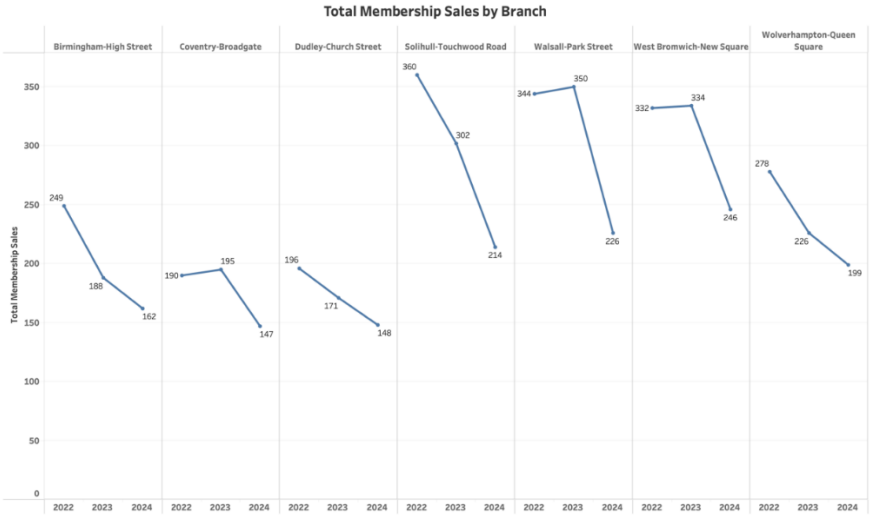
Captures branch-level quality ratings through average satisfaction scores tracking.

Insights & Actionability:

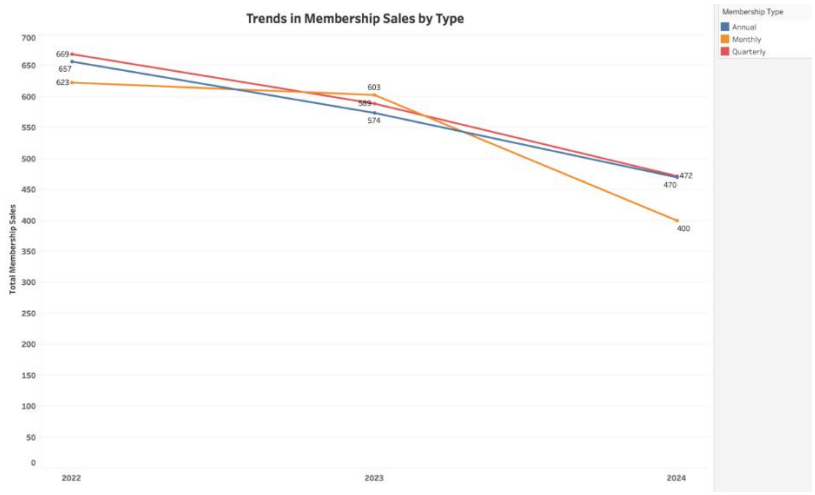
West-Bromwich significantly underperforms on average scores, while Dudley-Church Street remains top. Branch performance evaluations and corrective action plans are recommended.

5. Additional Metrics & Insights

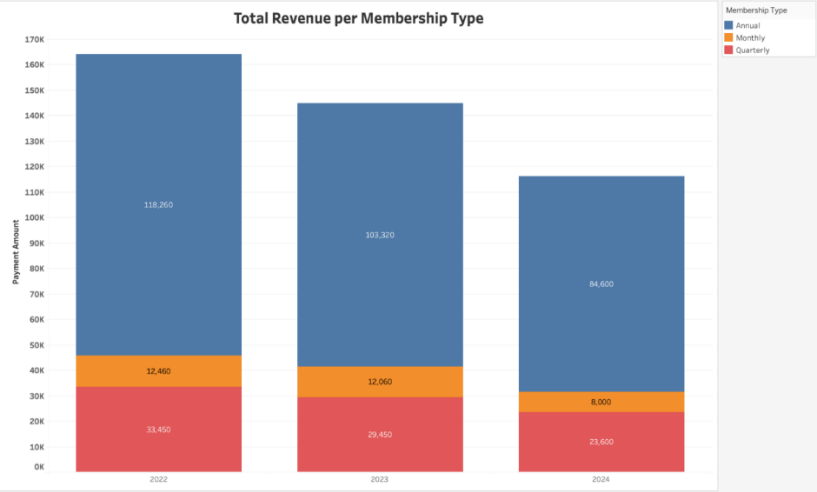
Total Memberships per Branch



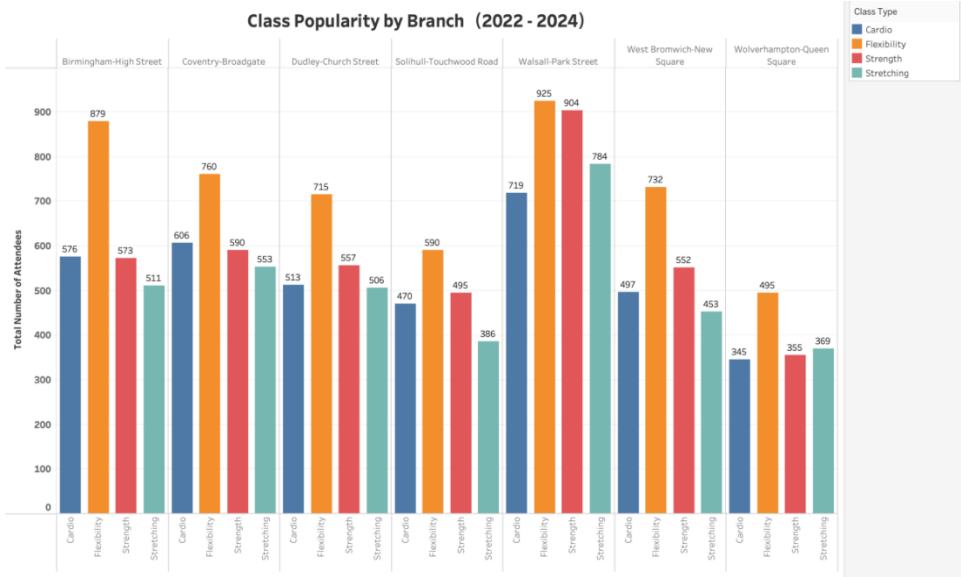
Total Members per Membership Type



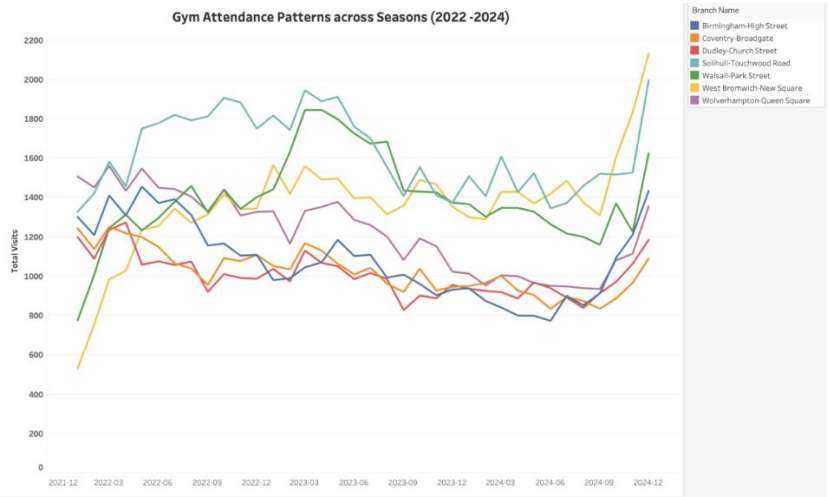
Total Revenue per Membership Type



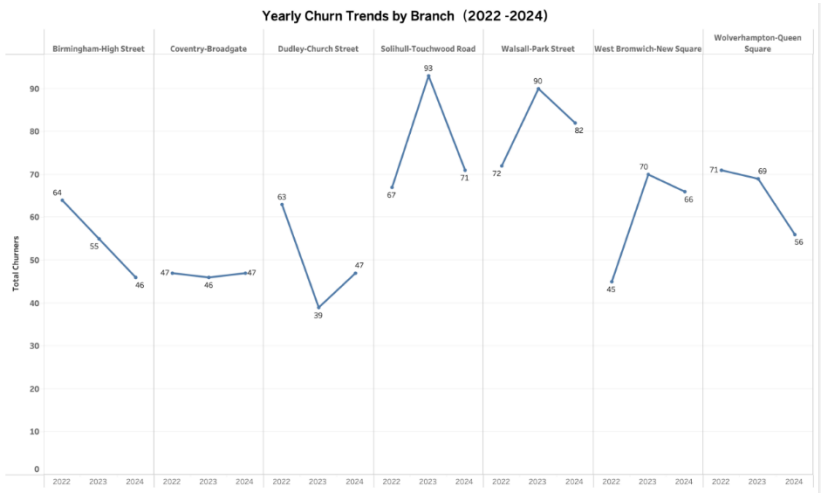
Class Popularity by Branch



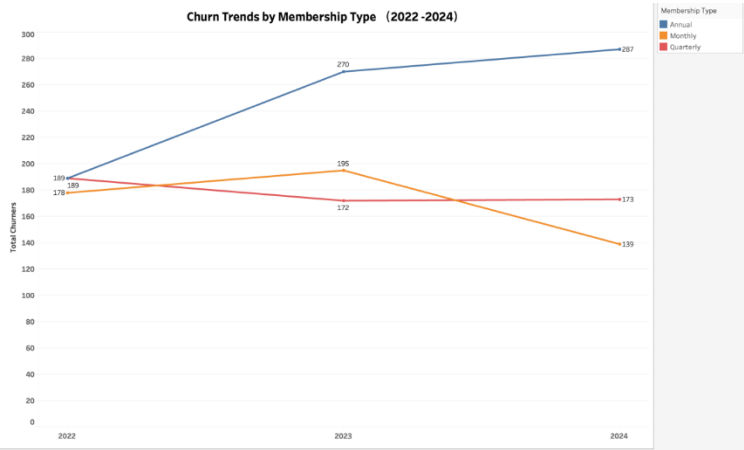
Gym Attendance Patterns across Seasons



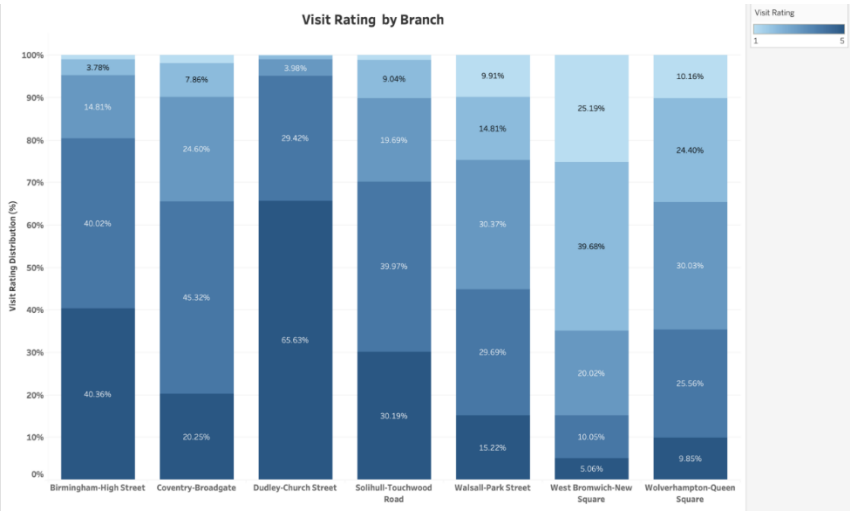
Churn Rate by Branch



Churn Rate by Membership Type



Visit Rating Split by Branch



Average Visit Rating by Branch