Wariant 1 - Referencyjna

Schemat bazy danych, walidacja

```
db.createCollection("companies", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "contactInfo"],
      properties: {
        name: {
          bsonType: "string",
          description: "Nazwa firmy"
        },
        description: {
          bsonType: "string"
        },
        contactInfo: {
          bsonType: "object",
          required: ["email", "phone"],
          properties: {
            email: {
              bsonType: "string",
              pattern: "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$"
            },
            phone: {
              bsonType: "string"
            },
            address: {
              bsonType: "object",
              properties: {
                street: { bsonType: "string" },
                city: { bsonType: "string" },
                postalCode: { bsonType: "string" }
              }
            }
          }
        },
        foundedYear: {
          bsonType: "int",
          minimum: 1900,
          maximum: 2025
        },
        licenseNumber: {
          bsonType: "string"
        }
      }
    }
  }
});
```

```
db.createCollection("tours", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["companyId", "title", "startDate", "endDate", "price"],
      properties: {
        companyId: {
          bsonType: "objectId",
          description: "Referencja do firmy organizującej wycieczkę"
        },
        title: {
          bsonType: "string",
          minLength: 5,
          maxLength: 100
        },
        description: {
          bsonType: "string"
        },
        startDate: {
          bsonType: "date"
        },
        endDate: {
          bsonType: "date"
        },
        price: {
          bsonType: "int",
          minimum: 0
        },
        capacity: {
          bsonType: "int",
          minimum: 1
        },
        reservationsCount: {
          bsonType: "int",
          minimum: 0
        },
        destination: {
          bsonType: "string"
        },
        itinerary: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["day", "description"],
            properties: {
              day: { bsonType: "int", minimum: 1 },
              description: { bsonType: "string" }
            }
          }
        },
        tags: {
          bsonType: "array",
```

```
          items: {
            bsonType: "string"
          }
        }
      }
    }
  }
});


db.createCollection("users", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["firstName", "lastName", "email"],
      properties: {
        firstName: {
          bsonType: "string"
        },
        lastName: {
          bsonType: "string"
        },
        email: {
          bsonType: "string",
          pattern: "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$"
        },
        phone: {
          bsonType: "string"
        },
        dateOfBirth: {
          bsonType: "date"
        },
        registrationDate: {
          bsonType: "date"
        }
      }
    }
  }
});


db.createCollection("reservations", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["userId", "tourId", "bookingDate", "status"],
      properties: {
        userId: {
          bsonType: "objectId",
          description: "Referencja do użytkownika"
        },
        tourId: {
          bsonType: "objectId",
          description: "Referencja do wycieczki"
```

```
        },
        bookingDate: {
          bsonType: "date"
        },
        status: {
          bsonType: "string",
          enum: ["confirmed", "pending", "cancelled"]
        },
        participantsCount: {
          bsonType: "int",
          minimum: 0
        },
        totalPrice: {
          bsonType: "int",
          minimum: 0
        },
        paymentStatus: {
          bsonType: "string",
          enum: ["paid", "pending", "partial"]
        },
        specialRequests: {
          bsonType: "string"
        }
      }
    }
  }
});


db.createCollection("reviews", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["userId", "tourId", "rating", "createdAt"],
      properties: {
        userId: {
          bsonType: "objectId",
          description: "Referencja do użytkownika"
        },
        tourId: {
          bsonType: "objectId",
          description: "Referencja do wycieczki"
        },
        rating: {
          bsonType: "double",
          minimum: 1,
          maximum: 5
        },
        comment: {
          bsonType: "string"
        },
        createdAt: {
          bsonType: "date"
        }
```

```
      }
    }
  }
});
```

Tworzenie danych (i ustalanie zmiennych pod testy)

```
const companyId = db.companies.insertOne({
  name: "Podróże Marzeń",
  description: "Biuro podróży specjalizujące się w wycieczkach egzotycznych",
  contactInfo: {
    email: "kontakt@podrozemarzen.pl",
    phone: "123456789",
    address: {
      street: "Turystyczna 10",
      city: "Warszawa",
      postalCode: "00-001"
    }
  },
  foundedYear: 2010,
  licenseNumber: "TUR/123/2010"
}).insertedId;


const tourId = db.tours.insertOne({
  companyId: companyId,
  title: "Wyprawa do Maroka",
  description: "Tygodniowa wycieczka objazdowa po Maroku",
  startDate: ISODate("2025-06-15"),
  endDate: ISODate("2025-06-22"),
  price: 3500,
  capacity: 20,
  reservationsCount: 2,
  destination: "Maroko",
  itinerary: [
    { day: 1, description: "Przylot do Marrakeszu, zakwaterowanie" },
    { day: 2, description: "Zwiedzanie Marrakeszu" },
    { day: 3, description: "Przejazd do Fez" }
  ],
  tags: ["Afryka", "zwiedzanie", "kultura"]
}).insertedId;


const tourId2 = db.tours.insertOne({
  companyId: companyId,
  title: "Safari w Kenii",
  description: "10-dniowe safari w parkach narodowych Kenii",
  startDate: ISODate("2025-07-10"),
  endDate: ISODate("2025-07-20"),
  price: 5000,
```

```
  capacity: 15,
  reservationsCount: 0,
  destination: "Kenia",
  itinerary: [
    { day: 1, description: "Przylot do Nairobi" },
    { day: 2, description: "Park Narodowy Amboseli" }
  ],
  tags: ["Afryka", "safari", "przyroda"]
}).insertedId;


const userId1 = db.users.insertOne({
  firstName: "Jan",
  lastName: "Kowalski",
  email: "jan.kowalski@example.com",
  phone: "987654321",
  dateOfBirth: ISODate("1985-04-12"),
  registrationDate: ISODate("2023-01-15")
}).insertedId;

const userId2 = db.users.insertOne({
  firstName: "Anna",
  lastName: "Nowak",
  email: "anna.nowak@example.com",
  phone: "123456789",
  dateOfBirth: ISODate("1990-07-22"),
  registrationDate: ISODate("2023-03-05")
}).insertedId;


const reservationId = db.reservations.insertOne({
  userId: userId1,
  tourId: tourId,
  bookingDate: ISODate("2025-01-10"),
  status: "confirmed",
  participantsCount: 2,
  totalPrice: 7000,
  paymentStatus: "paid",
  specialRequests: "Pokój z widokiem na morze"
}).insertedId;


const reviewId = db.reviews.insertOne({
  userId: userId1,
  tourId: tourId,
  rating: 4.5,
  comment: "Świetna organizacja i wspaniałe atrakcje!",
  createdAt: ISODate("2025-06-25")
}).insertedId;
```

Prezentacja wad i zalet:

ZALETA 1: Łatwość aktualizacji

Zmiana danych firmy - aktualizacja jednego dokumentu

```
db.companies.updateOne(
  { _id: companyId },
  {
    $set: {
      "contactInfo.email": "nowy.email@podrozemarzen.pl",
      "contactInfo.phone": "555888999"
    }
  }
);
```

Modyfikacja jest prosta i nie wymaga aktualizacji w wielu miejscach

ZALETA 2: Normalizacja danych

Pobieranie informacji o firmie - dane są przechowywane tylko raz

```
db.companies.findOne({ _id: companyId });
```

Dane kontaktowe firmy są tylko w jednym miejscu, co zapobiega niespójnościom

ZALETA 3: Elastyczność

Dodanie nowych pól do wycieczki bez wpływu na inne kolekcje

```
db.tours.updateOne(
  { _id: tourId },
  {
    $set: {
      difficultyLevel: "średni",
      requiredEquipment: ["wygodne buty", "nakrycie głowy", "krem z filtrem"],
      minAge: 12
    }
  }
);
```

Model można rozszerzać elastycznie bez wpływu na powiązane kolekcje

WADA 1: Konieczność wykonywania wielu zapytań do pobrania powiązanych danych

Pobranie informacji o rezerwacji wraz z danymi użytkownika i wycieczki

```
const reservation = db.reservations.findOne({ _id: reservationId });
const user = db.users.findOne({ _id: reservation.userId });
const tour = db.tours.findOne({ _id: reservation.tourId });
const company = db.companies.findOne({ _id: tour.companyId });
```

Potrzebujemy 4 oddzielnych zapytań, aby pobrać pełne informacje o rezerwacji

WADA 2: Złożone operacje wymagają skomplikowanych agregacji

Pobranie wszystkich rezerwacji użytkownika wraz z danymi wycieczek

```
db.reservations.aggregate([
  { $match: { userId: userId1 } },
  { $lookup: {
      from: "tours",
      localField: "tourId",
      foreignField: "_id",
      as: "tourDetails"
    }
  },
  { $unwind: "$tourDetails" },
  { $lookup: {
      from: "companies",
      localField: "tourDetails.companyId",
      foreignField: "_id",
      as: "companyDetails"
    }
  },
  { $unwind: "$companyDetails" }
]);
```

Wymagane jest złożone zapytanie, które nieprzyjemnie się pisze i czyta. Oraz nie jest najszybsze.

WADA 3: Mniejsza wydajność przy złożonych operacjach

Pobieranie średniej oceny dla każdej wycieczki danej firmy wymaga agregacji, która nie jest wydajna.

```
db.tours.aggregate([
  { $match: { companyId: companyId } },
  { $lookup: {
      from: "reviews",
      localField: "_id",
      foreignField: "tourId",
      as: "reviewsData"
    }
  },
  { $project: {
      title: 1,
```

```
        destination: 1,
        averageRating: { $avg: "$reviewsData.rating" },
        reviewCount: { $size: "$reviewsData" }
      }
    }
  ]);
```