Важное исправление в работе с SQLAlchemy

Невероятно ценная ремарка, которая уменьшит время на прохождение курса и сохранит часть нервов.

Как вы увидите дальше, во всех видео для возвращения результатов от Алхимии используется конструкция `result.all()` или `result.scalars().all()`, которые доставляют много проблем и дополнительных усилий при дальнейшем переводе ответа Алхимии к JSON формату. Просьба во всех случаях, где я использую конструкции выше, использовать конструкцию

```
result.mappings().all()
```

которая сразу же возвращает список словариков в виде `{столбец1: значение, столбец2: значение}`.

Пожалуйста, запомните это и применяйте везде по ходу курса!

P.S. Весь код в репозитории был переписан на `mappings()`.

Возвращение ответа для запросов с JOIN'ами

Если вы написали запрос через SQLAlchemy ORM, который возвращает столбцы из несколько таблиц при помощи JOIN, например,

```
# В этом запросе мы хотим получить все данные о бронированиях и номерах для конкретного пользователя
async with async_session_maker() as session:
    query = (
        select(
            Bookings,  # <-- первая таблица
            Rooms,  # <-- вторая таблица
        )
        .join(Rooms, Rooms.id == Bookings.room_id, isouter=True)
        .where(Bookings.user_id == user_id)
    )
    result = await session.execute(query)
    return result.mappings.all()
```

то вы увидите неожиданный ответ:

```
[
  {
```

```
    "Bookings": {
      "user_id": 6,
      "room_id": 3,
      "date_to": "2020-01-02",
      "price": 4570,
      "total_days": 1,
      "id": 24,
      "date_from": "2020-01-01",
      "total_cost": 4570
    },
    "Rooms": {
      "id": 3,
      "description": null,
      "services": [],
      "quantity": 15,
      "hotel_id": 2,
      "name": "Номер на 2-х человек",
      "price": 4570,
      "image_id": 9
    }
  }
]
```


хотя вы явно ожидали увидеть структуру без вложенности:


```
[
  {
    "id": 24,
    "room_id": 3,
    "user_id": 6,
    "date_from": "2020-01-01",
    "date_to": "2020-01-02",
    "price": 4570,
    "total_cost": 4570,
    "total_days": 1,
    "id_1": 3,
    "hotel_id": 2,
    "name": "Номер на 2-х человек",
    "description": null,
    "price_1": 4570,
    "services": [],
    "quantity": 15,
    "image_id": 9
  }
]
```

С таким объектом удобно работать и на бэкенде, и на фронтенде. Чтобы добиться такого ответа от Алхимии, достаточно добавить метод `__table__.columns` после указания названия модели:

```
async with async_session_maker() as session:
    query = (
        select(
            # __table__.columns нужен для отсутствия вложенности в ответе Алхимии
            Bookings.__table__.columns,
            Rooms.__table__.columns,
        )
        .join(Rooms, Rooms.id == Bookings.room_id, isouter=True)
        .where(Bookings.user_id == user_id)
    )
    result = await session.execute(query)
    return result.mappings().all()
```

+1