**Итоговое задание по курсу БД. 2 семестр.**

Требуется написать backend и клиентское приложение (web или desktop) для работы с предметной областью в соответствии с вариантом, где модель предметной области реализуется в виде SQL-DB.

CRUD = Create, Read, Update, Delete.

Общие для вариантов 1, 2, 3, 42-48 функциональные возможности:

1. Аутентификация + авторизация пользователя + восстановление пароля.
2. CRUD всех сущностей в БД / предметной области через DataGrid-ы по значимым модельным таблицам (т.е. не учитывать связующие таблицы; реализация только через ORM).  
   Примечание: максимально использовать стандартные компоненты; предусмотреть возможность оптимальной работы с большими данными и частыми однотипными запросами (Cache service; пагинация; …).
3. Задание в конфиге Connection string (IP, порт).
4. Ввод и исполнение запросов вручную.
5. [Опционально, желающим получить 5] Модуль управления пользователями системы (CRUD), включающий в себя ролевую систему по принципу ACL-таблицы.  
   Примечание: права, роли и связи между ними могут быть жёсткими, но содержаться должны не хардкорно, а в конфигах. Прав можно реализовать штуки 3-4 несложных.

Нефункциональные требования:

1. Понятный графический (!) интерфейс (никаких ручных запросов (за исключением п. 4 выше)), сделайте его модульным (разбить его на разделы / модули в соответствии с необходимой функциональностью).  
   Важно: уделите особое внимание проектированию интерфейса, продумайте его ДО начала кодирования, а НЕ в процессе.
2. Написать клиент на любом ОО-языке (С++, Java, C#; технологии ASP.NET MVC (очень приветствуется ☺) \ jQuery \ AJAX \ Bootstrap \ React; крайне не желательно писать на Python или Perl).
3. При написании клиента использовать принципы ОО-проектирования и SOLID, архитектуру делать слабосвязной и масштабируемой в соответствии с MVC-паттерном. Однако особо будет оценён SOA / mSOA-подход.

За основу БД, с которой будет осуществляться работа, взять свой вариант ниже.

**Вариант 1. Ж/Д транспорт.**

Схема БД, соответствующая заданию 2.1-2.2.

Необходимо продумать UI-интерфейсы и реализовать следующие функциональные возможности в клиенте:

1. Просмотр, фильтрация и редактирование расписаний (поездок) движения поездов, в т.ч. задержек в текущем расписании и числу свободных мест (билетов) по типам на каждом участке маршрута.  
   Примечание: фильтр (поиск) + общий список расписаний между городами + профиль конкретной поездки (включает поезд, маршрут и т.д.).  
   Фильтры должны включать как минимум поиск (допустимы любые комбинации данных фильтров):
   1. По городам поездок.
   2. По датам (отправления, прибытия).
   3. По поездам (номерам).
   4. Наличию задержек + диапазон задержек.
   5. Другое (проявить фантазию).
2. Генерация новых поездок поезда на основе последней поездки (учесть числа месяца в предыдущих поездках).
3. CRUD поездов с набором фильтров (фильтр (поиск) + общий список поездов + профиль конкретного поезда).  
   В профиле поезда должны быть доступны для редактирования как минимум:
   1. Число и типы мест.
   2. Поездная бригада.
   3. Маршрут.
   4. Переход в блок работы с расписанием данного поезда (предзаполнение и применение фильтра).
4. Просмотр и редактирование маршрутов и привязанных поездов с фильтрами, как минимум, по городам.
5. Управление сотрудниками РЖД, поездными бригадами и связанными с ними поездами.  
   Примечание: учитывать график движения поездов при формировании бригады.
6. Покупка / возврат билета заданного типа на поезд в заданное время между заданными городами с валидацией корректности (продумать UI).
7. Поиск ближайших поездов / поездок в указанный город в указанный отрезок времени.  
   Должны быть доступным сортировки: по времени в пути между заданными станциями; по времени отправления со станции отбытия; по количеству станций в маршруте.
8. Дополнение п. 7: просмотр полного маршрута движения конкретного поезда в заданное время с датами-временами прибытия на каждую станцию.

**Важно:** максимально следить за соблюдением целостности (к примеру, поезд не может прибывать на 3-ю станцию маршрута раньше, чем на 2-ую и т.д.).

**Вариант 2. Банк (АБС).**

Схема БД (которую нужно спроектировать) и приложение для работы оператора банка должны учитывать следующие функциональные возможности:

1. CRUD по тарифам (условиям доступных кредитов: доступные сроки, ставки, суммы).
2. Блокировка клиента на заданный период.
3. Присвоение клиенту тарифа (одобрение кредита с условиями, соответствующими тарифу).
4. Создание кредитов с аннуитетными платежами (равные платежи через равные отрезки времени, формулу расчёта аннуитета по заданным (в интерфейсе) телу кредита, сроку и ставке легко загуглить).
5. Внесение платежа по кредиту (оператором вручную) с учётом каналов: через банковский счёт (комиссии нет), карту (есть комиссия за выдачу (падает в долг)), Yandex.Money (есть комиссия), QIWI (есть комиссия).  
   Важно: при внесении платежа стоит учитывать кейсы (типы платежей):
   1. Досрочный платёж => пересчёт оставшейся части Графика платежей с учётом аннуитентности.
   2. Платёж в дату по Графику платежей, но не полное погашение кредита.
   3. Полное погашение кредита в График (последний платёж; кредит надо закрыть).
   4. Полное досрочное погашение кредита (рассчитывать необходимую сумму; кредит нужно закрыть).
   5. Сумма платежа больше необходимой по расчётам (остаток падает в переплату).
6. CRUD по клиентам-кредитам с поиском (фильтрами) и профилем клиента-кредита.  
   Важно: клиент может иметь в один момент времени не более одного активного кредита.  
   Профиль клиента включает (разбить на (под)вкладки):
   1. Вся персональная информация.
   2. Вся история кредитов в системе.
   3. График платежей по активному кредиту (если есть).
   4. Наличие просрочки и информации по ней.
   5. Балансы по кредиту: тело кредита, проценты, штрафы, комиссии за выдачу.
   6. Все платежи (выдача; внесённые платежи, CRUD этих платежей) по различным каналам выдачи / погашения.

Фильтры включают в себя, как минимум:

* + - Все параметры профиля кредита или клиента.
    - Состояние кредита (закрыт, активен, просрочен).
    - Состояние клиента (нет кредитов / есть тариф; есть кредит; нет тарифа; заблокирован).
    - Текущая прибыльность: диапазон от и до.  
      Прибыльность = <входящий денежный поток по кредиту на момент <сейчас>> / <тело кредита>

1. Отчёт с агрегацией последовательно по <дням => месяцам => годам> (см. ROLLUP) по выдачам с указанием следующих данных:
   1. Объём выдачи за период.
   2. Объём возвратов за период.
   3. Процент просроченных кредитов на момент времени <текущий день> с датой погашения, попадающий в заданный период.
   4. Итоговая прибыльность (<все платежи по кредитам до момента <сейчас> с датой погашения, попадающей в период> / <сумма тел кредитов с датой погашения, попадающей в период>) на момент <текущий день>.
   5. Прибыльность +2M на момент времени <дата погашения по договору + 2 месяца>.

**Важно**: максимально предусмотреть все ограничения целостности (к примеру, нельзя создать кредит у клиента с уже существующим кредитом или клиенту без тарифа или с тарифом, но с несоответствующими условиями; невозможно внести платёж по несуществующему кредиту и т.д.).

**Вариант 3. Система регистраций на спортивные мероприятия (+1 балл за реализацию на .NET-е :))**

Прототип цели: <https://n445800.yclients.com/company:422607/activities> - по сути надо создать копию данной системы регистрации и оплаты спортивных тренировок.

Общие требования:

1. БД – Postgres SQL, MySQL.
2. Использование ORM (EF).
3. Использование архитектуры MVC + шаблонизатора (Razor).
4. Опционально: можно взять за основу черновик проекта: <http://inforsec.ru/bd/EventRegSystem.zip> - не обязательно.

Функционал Администратора:

1. CRUD (создать, редактировать, посмотреть, удалить) «Спортивных площадок» в ЛК, атрибуты:

* Название, адрес, макс. количество мест.
* Дефолтные параметры мероприятий на площадке: стоимость, длительность, число мест, тренер.

1. CRUD спортивных мероприятий, проводимых на площадках, атрибуты:

* Время начала, стоимость, длительность, число мест, тренер (из списка тренеров).
* CRUD регистраций (посетителей) на мероприятие.

1. Фильтры спорт. мероприятий по атрибутам.
2. Просмотр и CRUD регистраций посетителей на спорт. мероприятия, функционал подтверждения регистрации.
3. CRUD посетителей (+ фильтры по атрибутам).
4. Отчёты по посещаемости площадок с фильтрами по датам, площадкам.

Функционал Посетителя:

1. Просмотр спорт. мероприятий в соответствии с фильтрами и хронологией: сокращённо и полно (со всеми атрибутами, в т.ч. общее число мест, число свободных мест).
2. Регистрация посетителя на мероприятие (с проверкой, есть ли такой посетитель (ФИО, телефон, e-mail): если да, спрашиваем пароль, если нет, регистрируем): после регистрации – e-mail-уведомление на почту посетителя о регистрации.
   1. После подтверждения регистрации администратором – e-mail-уведомление о подтверждении.
3. E-mail уведомления всех записанных посетителей за N часов до даты-времени мероприятия (N – параметр в конфиге).

**Варианты 42-48** можно найти здесь: <http://inforsec.ru/bd/DB_projects.zip>