|  |  |
| --- | --- |
|  | **Технически университет - София**  **Филиал Пловдив** |

**Дипломна работа**

|  |  |  |
| --- | --- | --- |
| Тема: | ***Информационна система и база данни за системна администрация и инвентаризация на ИТ оборудване*** | |
|  | |  | |
| Студент**:** | | ***Иван Иванов Иванов*** | |
| Фак №: | | ***123456*** | |
| Специалност: | | ***КСТ*** | |
| Образователнастепен: | | ***БАКАЛАВЪР*** | |
| Факултет: | | ***ФЕА*** | |
| Ръководител: | | ***инж. Николай Николов*** | |

ТУ – София, Филиал Пловдив, 2017 г.

СЪДЪРЖАНИЕ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Увод** | …………………………………………………………… | | | | 5 |
| **Глава 1.** | Структура на постъпващата информация…………….. | | | | 7 |
| **Глава 2.** | Проектиране на структурата на базата данни............... | | | | 9 |
| **2.1.** | | Съставяне на ER модел на базата данни…………… | | | 9 |
| **2.1.1.** | | | Описание на entity типовете……………………... | | 9 |
| **2.1.2.** | | | Описание на типовете връзки……………………. | | 15 |
| **2.1.3.** | | | Обобщение на типове същности и връзки в таблица…………………………………………….. | | 20 |
| **2.1.4.** | | | Обобщен ER модел на предметната област…….. | | 21 |
| **2.2.** | | Описание на релационният модел на базата данни.. | | | 22 |
| **2.2.1.** | | | Таблици с имена на таблици, колони, типове данни иописание…………………………………. | | 24 |
| **2.2.2.** | | | Релационни таблици и връзките между тях | | 43 |
| **Глава 3.** | Обосновка за избор на машина за база данни, език за програмиране и програмна среда……………………... | | | | 48 |
| **3.1.** | | Обосновка за избор на машина за бази данни…….. | | | 48 |
| **3.2.** | | Обосновка за избор на език за програмиране……... | | | 49 |
| **3.3.** | | Обосновка за избор на програмна среда…………… | | | 50 |
| **Глава 4.** | Описание на основните програмни модули и на взаимодействието между тях………………………….. | | | | 51 |
| **4.1.** | | Описание на форма модул за данни DModFRM…... | | | 51 |
| **4.1.1.** | | | Цел и логика на формата…………………………. | | 52 |
| **4.1.2.** | | | Съставни части на формата……………………… | | 53 |
| **4.1.3.** | | | Събития във формата……………………………... | | 54 |
| **4.2.** | | Описание на главна форма IBSysAdminMainFRM... | | | 54 |
| **4.2.1.** | | | Цел и логика на формата…………………………. | | 54 |
| **4.2.2.** | | | Съставни части на формата……………………… | | 55 |
| **4.2.3.** | | | Събития във формата……………………………... | | 56 |
| **4.3.** | | Описание на форма за редактиране на компонентиComponentEditFORM….......…………... | | | 57 |
| **4.3.1.** | | | Цел и логика на формата…………………………. | | 57 |
| **4.3.2.** | | | Съставни части на формата……………………… | | 58 |
| **4.3.3.** | | | Събития във формата……………………………... | | 59 |
| **4.4.** | | Описание на форма за справки RefFORM…………. | | | 60 |
| **4.4.1.** | | | Цел и логика на формата…………………………. | | 60 |
| **4.4.2.** | | | Съставни части на формата……………………… | | 60 |
| **4.4.3.** | | | Събития във формата……………………………... | | 61 |
| **4.5.** | | Описание на форма за служители PersonFORM…... | | | 62 |
| **4.5.1.** | | | Цел и логика на формата…………………………. | | 62 |
| **4.5.2.** | | | Съставни части на формата……………………… | | 63 |
| **4.5.3.** | | | Събития във формата……………………………... | | 64 |
| **4.6.** | | Описание на форма за редактиране на офиси OfficeFORM………………………………………….. | | | 64 |
| **4.6.1.** | | | Цел и логика на формата…………………………. | | 64 |
| **4.6.2.** | | | Съставни части на формата……………………… | | 65 |
| **4.6.3.** | | | Събития във формата……………………………... | | 65 |
| **4.7.** | | Описание на форма за дейности ActRecFORM…… | | | 66 |
| **4.7.1.** | | | Цел и логика на формата…………………………. | | 66 |
| **4.7.2.** | | | Съставни части на формата……………………… | | 66 |
| **4.7.3.** | | | Събития във формата……………………………... | | 67 |
| **4.8.** | | Описание на форма за редактиране на номенклатурапроизводители ManufactFORM……. | | | 67 |
| **4.8.1.** | | | Цел и логика на формата…………………………. | | 67 |
| **4.8.2.** | | | Съставни части на формата……………………… | | 68 |
| **4.8.3.** | | | Събития във формата……………………………... | | 68 |
| **4.9.** | | Описание на форма за редактиране на номенклатураотдели DepFORM…………………… | | | 69 |
| **4.9.1.** | | | Цел и логика на формата…………………………. | | 69 |
| **4.9.2.** | | | Съставни части на формата……………………… | | 70 |
| **4.9.3.** | | | Събития във формата……………………………... | | 70 |
| **4.10.** | | Описание на форма за редактиране на номенклатурадлъжности PosFORM……………….. | | | 70 |
| **4.10.1.** | | | Цел и логика на формата…………………………. | | 70 |
| **4.10.2.** | | | Съставни части на формата……………………… | | 71 |
| **4.10.3.** | | | Събития във формата……………………………... | | 71 |
| **4.11.** | | Описание на форма за редактиране на номенклатуратип на дейност ActTypeFORM……... | | | 72 |
| **4.11.1.** | | | Цел и логика на формата…………………………. | | 72 |
| **4.11.2.** | | | Съставни части на формата……………………… | | 72 |
| **4.11.3.** | | | Събития във формата……………………………... | | 73 |
| **Глава 5.** | Ръководство за потребителя…………………………... | | | | 74 |
| **5.1.** | | Добавяне, изтриване и редактиране на компютър… | | | 74 |
| **5.2.** | | Добавяне, изтриване и редактиране на компонент... | | | 75 |
| **5.3.** | | Добавяне, изтриване и редактиране на принтер…... | | | 76 |
| **5.4.** | | Добавяне, изтриване и редактиране на комутатор | | | 76 |
| **5.5.** | | Добавяне, изтриване и редактиране на маршрутизатор………………………………………. | | | 76 |
| **5.6.** | | Добавяне, изтриване и редактиране на UPS устройство…………………………………………… | | | 76 |
| **5.7.** | | Добавяне, изтриване и редактиране на служители... | | | 76 |
| **5.8.** | | Добавяне, изтриване и редактиране на събитие от дневника……………………………………………… | | | 77 |
| **5.9.** | | Редактиране на номенклатурите в базата данни…... | | | 77 |
| **5.10.** | | Съставяне на справки……………………………….. | | | 77 |
| **Използвана литература** | | | | ……………………………………………... | 78 |
| **Приложение** | Табло със структурата на базата данни………………. | | | | 79 |
| **Приложение** | Основни програмни модули и взаимовръзките между тях……………………………………………………….. | | | | 80 |
| **Приложение** | SQL скрипт на базата данни…………………………... | | | | 81 |
| **Приложение** | Сорс код на програмното осигуряване……………….. | | | | 94 |

**Увод**

В процесът на създаване на една информационна управляваща система, включваща в себе си база от данни, се преминава през определени етапи. Това са етапите на проектиране на базата данни и проектиране на програмното осигуряване, което ще работи с нея. Процесът на проектиране на такъв тип приложение представлява последователно изпълнение на определени процедури, които целят пълният анализ и оценка на нуждите на бъдещото програмно осигуряване. То, от своя страна, ще посрещне нуждите на потребителите, които ще работят с него. Всяко едно такова приложение, само по себе си, представлява отделен инженерен проблем, изискващ своят анализ, проектиране и разработка. Това е така, защото нуждите, наложени от бъдещите потребители, както и самата среда на приложение на съответното програмно осигуряване са винаги различни.

Целта на настоящата дипломна работа е да разработи приложение, което представлява информационна управляваща система с бази данни за системна администрация и инвентаризация на ИТ оборудване. Тази система има за цел да разреши проблема, който стои пред всяка една ИТ компания, а именно – организация и каталогизиране на дейностите по системната администрация, и управление на наличното оборудване. Етапите за разработване на приложното програмно осигуряване са:

1. Анализ и събиране на изискванията към базата с данни и приложението – за конкретното приложение изискванията към данните и операциите са определени.
2. Функционален анализ и концептуален дизайн на базата данни – определят се бъдещите трансакции и концептуалния модел на базата данни на високо ниво. За концептуален модел в конкретната задача е използван ER моделът, а бъдещите трансакции са имплементирани в самото приложение на С++ Builder.
3. Логически дизайн – определя се моделът на данните. Тук вече се прилага конкретната машина за бази данни и концептуалният модел се транслира в модел, който имплементира модела на данните. Настоящото приложение използва InterBase Serverv6.5 (като машина за бази данни) и релационният модел за описване на структурата на БД.
4. Физически дизайн – определя организацията на файлове, индекси и други физически параметри. В конкретната задача се реализира от машината за бази данни.

Разработеното приложение е от тип клиент-сървър, което може да се използва като Desktop или Intranet приложение, работещо под ОС MS Windows. Целта на това програмно осигуряване е да опише всички характеристики на предметната област, като ги систематизира като данни, пригодни за съхраняване в база данни, които, от своя страна, могат да бъдат манипулирани по-късно по познати методи и техники.

Осигуреното програмно осигуряване предлага и запис на направената справка в текстов файл с цел отпечатване. По този начин може да се извеждат специални справки, които биха послужили като официален документ към архива на ИТ компанията, отнасящ се за определени наличности, на определени компоненти или устройства, в даден период от време. Така може да се създаде и справочна история, която да спомогне в проследяването на дейността на ИТ фирмата, както и следенето на определени тенденции.

**Глава 1**

**Структура на постъпващата информация**

Структурата на постъпващата информация се определя от конкретната област, в която се прилага приложението и базата данни. В конкретната задача основните единици информация представляват дейности, които се извършват по администрацията и инвентаризацията на ИТ оборудването и сакакто следва:

* Обновяване на оборудване
* Профилактика на оборудване
* Бракуване на оборудване
* Вписване на извършените дейности в дневник
* Насрочване на дейности посредством календар
* Извеждане на списъци с наличното оборудване във фирмата

За да може бъдещото приложение да обхване всеки един аспект от тези дейности, в базата от данни се пази информация за:

* Извършени дейности
* Предстоящи дейности
* Компютри и техните компоненти
* Активно мрежово оборудване
* Мултифункционални устройства

На базата на наличната информация могат да се правят множество справки:

* за бракуване, обновяване, профилактика, както и за това дали дадена дейност е изминала или все още предстои
* за определен период от време на дейността
* за определени компютърни конфигурации – с определени компоненти, производител, офис и т.н.
* за определени мултифункционални устройства – с определен период на бракуване, изтичане на гаранцията или датана фактурата
* за определени активни мрежови устройства – с определени технически характеристики или срокове
* за конкретни компоненти и техните конкретни технически характеристики
* за офиси – с определено местоположение или име
* за служители във фирмата –с определено име, презиме, фамилия, ЕГН или телефон номер

**Глава 2**

**Проектиране на структурата на базата данни**

**2.1. Съставяне на ER модел на базата данни**

Първият етап от построяването на концептуалния модел на базата данни е съставянето на Entity Relationship модел. Най-напред се определят entity типовете (entity types)от предметната област на приложение на базата данни. След това се определят различни смислови връзки (relationship types), които съществуват между entity-тата, като всяка една бинарна връзка се характеризира с ограниченията кардниналност (cardinality ratio)и степен на участие (participation). Те се наричат още структурни ограничения (structural constraints).

**2.1.1. Описание на типовете същности (entity types)**

За тази задача предметната област на приложение на базата данни се състои от множество устройства, служители и отдели, при което всички те принадлежат на една организация, но на различни географски местоположения. Да разгледаме конкретно всеки един entity тип, като обясним всички негови характеристики, които са нужни за конкретното приложение на базата данни. В същото време ще бъде илюстрирано даденоentity чрез примитивите и средствата на ER моделирането, за да се покаже какво на практика е ER моделът. Устройства от предметната област:

* Мултифункционални устройства – както е добре известно, едно мултифункционално устройство може да включва в състава си принтер, скенер, копир и факс. За нуждите по системната администрация и инвентаризация ще се използва само принтерът. Entity тип ***Принтер*** включва следните атрибути:
  + Производител
  + Модел
  + Страници/минута
  + Дата на фактура
  + Номер на фактура
  + Крайна дата на гаранция
  + Дата на бракуване
  + Офис и неговото местоположение
* UPS устройства – това са устройствата, които временно осигуряват електрозахранване и служат като акумулатори при повреда в общата електрическа мрежа. Като такива, те са абсолютна необходимост за една ИТ фирма. Entity ***UPS***има следните атрибути:
  + Производител
  + Модел
  + Максимален брой станции, които могат да се захранват
  + Максимален брой минути на електрозахранване
  + Дата на фактура
  + Номер на фактура
  + Крайна дата на гаранция
  + Дата на бракуване
  + Офис и неговото местоположение
* Активно мрежово оборудване – това е мрежовото оборудване, което е активно и участва в управлението на мрежовия трафик, като взима решения относно неговото маршрутизиране и изолиране. Представители на този вид оборудване в конкретната задача са комутаторът (switch)и маршрутизаторът (router). Долу са описани самите Entity типове:
  + Entity ***Комутатор*** (***switch***)има следните атрибути:
    - Производител
    - Модел
    - Скорост (Gbit/s)
    - Общ брой портове
    - Дата на фактура
    - Номер на фактура
    - Крайна дата на гаранция
    - Дата на бракуване
    - Офис и неговото местоположение
  + Entity ***Маршрутизатор*** (***router***)има следните атрибути:
    - Производител
    - Модел
    - Максимален брой поддържани станции
    - Обхват на сигнала (м) – само ако е безжичен
    - Дата на фактура
    - Номер на фактура
    - Крайна дата на гаранция
    - Дата на бракуване
    - Офис и неговото местоположение
* Компютърни конфигурации – това са конфигурациите, които включват в състава си множество компоненти със свои собствени характеристики. Това разнообразие на различни хардуерни компоненти налага обособяването на всеки един такъв в отделен entity тип, за да може да се опишат конкретните за него характеристики. Ето защо освен описанието на Entity компютър, тук ще се включи и описанието на отделните Entity-та, представляващи различни компоненти:
  + Entity ***Компютър*** има следните атрибути:
    - Тип (десктоп, сървър, лаптоп и т.н.)
    - Производител
    - Модел
    - Описание на наличните интерфейси
    - Дата на фактура
    - Номер на фактура
    - Крайна дата на гаранция
    - Дата на бракуване
    - Офис и неговото местоположение
    - Потребител-собственик

КОМПЮТЪР

**Фигура 1**

На фиг. 1 е показан начинът за изобразяване на Entity типът компютър и всички негови атрибути. Entity типовете се изобразяват в правоъгълници, а атрибутите в елипси. Както ще стане ясно по-късно, това не е окончателният модел на entity тип компютър, защото при следващите ревизии на модела част от атрибутите се обособяват в отделни entity типове или връзки (relationships). В следствие на факта, че всеки един конкретен компонент има атрибути, които са общи и за другите компоненти, се обособява нов entity тип, наречен ***Компонент***. Атрибутите му са както следва:

* + Entity ***Компонент*** има следните атрибути:
    - Тип (процесор, твърд диск и т.н.)
    - Дата на гаранция
    - Дата на фактура
    - Номер на фактура
    - Сериен номер на компонента
  + Entity ***Процесор*** има следните атрибути:
    - Производител
    - Модел
    - Тактова честота (MHz)
    - Брой ядра
    - Кеш памет (KB)
  + Entity ***Графична карта*** има следните атрибути:
    - Производител
    - Модел
    - Работна честота (MHz)
    - Капацитет на паметта (MB)
    - Брой ядра
  + Entity ***Твърд диск*** има следните атрибути:
    - Производител
    - Модел
    - Форм фактор
    - Капацитет (GB)
    - Кеш памет на електронната част (KB)
    - Обороти/минута
  + Entity ***Оптично устройство*** има следните атрибути:
    - Производител
    - Модел
    - Скорост на запис (52х, 48х и т.н.)
    - Скорост на четене (52х, 48х и т.н.)
    - Разпознавани формати
  + Entity ***LAN карта*** има следните атрибути:
    - Производител
    - Модел
    - Скорост (Gbit/s)
  + Entity ***RAM памет*** има следните атрибути:
    - Производител
    - Модел
    - Капацитет (MB)
    - Работна честота (MHz)
  + Entity ***Монитор*** има следните атрибути:
    - Производител
    - Модел
    - Размер на екрана (“)
  + Entity ***Други*** има следните атрибути:
    - Производител
    - Описание
* Служители – това са хората, работещи в дадена организация. Те могат да имат различни длъжности, отдели, местоработи и т.н. В ER модела на базата данни Entity ***Служител*** има следните атрибути:
  + Име
  + Презиме
  + Фамилия
  + ЕГН
  + Телефонен номер
  + Електронна поща
  + Длъжност
  + Отдел
* Дневник и календар на дейностите – това представляват всички дейности, извършени във ИТ фирмата, както и тези, които са планирани и предстои да се извършат. Entity ***Дневник*** служи и като дневник за изминали събития, и като календар за предстоящите. Този тип същност има следните атрибути:
  + Тип на дейността (бракуване, обновяване и т.н.)
  + Дата на дейността
  + Описание на дейността

На пръв поглед, това са всички entity типове от предметната област. Но при проектирането на Entity Relationship модела на дадена база данни може да се наблюдава повторяемост на определени атрибути в самите типове същности. Така, на базата на тези общи атрибути могат да се обособят нови entity типове (entities), чиито атрибути представляват общите. По този начин даден атрибут може да бъде повишен до отделно entity или дадено entity може да бъде понижено до атрибут, ако е свързано само с един entity тип. В конкретният модел лесно се забелязва повторяемостта на определени атрибути. Например, измежду всички компоненти, компютри, маршрутизатори, комутатори, UPS устройства и принтери присъства атрибутът *производител*. Веднага се стига до извода, че атрибутът производител може да стане отделно entity, като по-този начин се избягва повторяемостта и излишъкът на данни в самата база.На базата на този принцип се обособяват следните entity типове: *тип на дейност*, *тип на компонент, тип на компютър, град*, *община*, *област*, *отдел, офис* и *длъжност*. Описанието на всеки един entity тип от гореизброените е както следва:

* Атрибути на entity ***Производител***:
  + Име на производител
* Атрибути на entity ***Тип на дейност***:
  + Наименование на дейност
* Атрибути на entity ***Отдел***:
  + Име на отдел
* Атрибути на entity ***Длъжност***:
  + Наименование на длъжността
* Атрибути на entity ***Град***:
  + Община, в която се намира
  + ЕКАТТЕ – идентификатор на селището
  + Име на град
* Атрибути на entity ***Община***:
  + Наименование на община
  + Област, в която се намира
* Атрибути на entity ***Област***:
  + Име на областта
* Атрибути на entity ***Офис***:
  + Град, в който се намира офиса
  + Име на офиса
  + Адрес
* Атрибути на entity ***Тип на компютър***:
  + Тип
* Атрибути на entity ***Тип на компонент***:
  + Тип

Всички тези новополучени типове изпълняват ролята на номенклатури т.е. в бъдещата информационна управляваща система те ще променят съдържанието си рядко и към тях ще бъдат направени множество референции, които решават проблема с излишъкът на данни (data redundancy).

Това е съвкупността от всички Entity типове в предметната област на приложение на базата данни. Както беше споменато по-рано, всички entity типове могат да се разглеждат с много повече атрибути и детайли, но за конкретната задача и цел тези са напълно достатъчни и изчерпателни.

**2.1.2. Описание на типовете връзки (relationship types)**

Другият основен компонент, който изгражда ER модела на предметната област на дадена база данни е съвкупността от различните смислови връзки, които съществуват между отделните entity типове.С цел разясняване на процеса на проектиране на базата данни, те ще бъдат накратко обяснени.

Когато се определят entity типовете, връзките се категоризират най-напред като атрибутина различните entity-та. След няколкократен итеративен процес на „рафиниране“ на модела, те се обособяват като отделни компоненти, също резултат от ER моделирането. Както всеки един entity тип си има своите характеристики, наречени атрибути, така и всяка връзка си има своите атрибути и отличителни черти. Те састепен (degree of relationship) и структурни ограничения (structural constraints). Под степен на връзка се разбира броят на entity типовете, които тя свързва, като се отличават бинарни, тринарни и т.н. Структурните ограничения биват два вида (за бинарните връзки): cardinalityratio constraint и participation constraint:

* ограничение „кардиналност“ – то представлява максималният брой инстанции на типа връзка (relationship type), в които дадено entity може да участва
* ограничение „участие“ – то определя дали съществуването на дадено entity зависи от участието му във даден тип връзка, като се дефинира като минималният брой връзки, в които entity-то трябва да участва. Ограничението „участие“ се разделя на две подгрупи:
  + тотално участие (existence dependency) – показва, че всяка една инстанция (проявление) на дадено entity задължително трябва да участва в съответният тип връзка
  + частично участие (partial participation) – само част от множеството от инстанции на съответният entity тип участва в тази връзка, но не е задължително да са всички проявления

С представените дотук характеристики на моделирането на връзките в ER модела вече може да се говори как са проектирани връзките между отделните entity типове в приложението за системна администрация и инвентаризация на ИТ оборудване. За да се внесе по-голяма яснота, всяка една по-значима връзка или съвкупност от няколко връзки ще се илюстрират и обясняват по отделно.

Една от най-важните връзки в предметната област за конкретната задача е връзката компютър-компонент. Тя е наречена „*съдържа*“, с което се показва принадлежността на даден компонент към съответния компютър. Едно от основните ограничения тук е това, че едно проявление на entity компютър може да съдържа множество компоненти т.е. да участва в множество инстанции на типа връзка „*съдържа*“. Но едно проявление на entity компонент може да участва само в една инстанция на тип връзка „съдържа“. Това ограничение произхожда от самата предметна област, като на ниво ER модел се отразява чрез изобразяването на структурните ограничения. Те се записват в ER модела като 1, M или N до ъглите на ромба, който представя самата връзка. Ето как изглежда връзката „*съдържа*“ между entity типовете компютър и компонент:

КОМПЮТЪР

КОМПОНЕНТ

СЪДЪРЖА

**Фигура 2**

1

N

От фигура 2 става ясно, че entity *компютър* няма директна връзка с конкретен компонент, а е свързан с entity *компонент*. То представлява entity тип, който е обособен на базата на общите характеристики на всеки един конкретен компонент (процесор, видео карта, твърд диск и т.н.). На по-късен етап ще стане ясно, че по-този начин справките върху голям брой компоненти (и по-точно техните общи характеристики) ще се изпълняват по-бързо и ефективно. От тук следва, че entity-то *компонент* ще има връзка с всеки един конкретен компонент. Така информацията за даден компонент, както ще стане ясно в следващият етап от разработката на дипломната работа, ще се съхранява в две отделни таблици – една за конкретните технически (която е и номенклатура) характеристики на компонента и втора за общите. Връзката между entity *компонент* и съответният компонент може да се именува като „е“ или като „може да бъде“, защото, всъщност, винаги става въпрос за компонент, но за различни негови характеристики. В крайна сметка инженерното решение се интересува от бързодействието и ефективността, а то се определя в зависимост от това по какъв начин потребителите възнамеряват да ползват базата данни. Връзката между конкретните компоненти и типа същност *компонент* изглежда така:

**Фигура 3**

1

ВИДЕО КАРТА

Е

1

ОПТИЧНО У-ВО

Е

N

N

N

N

N

N

N

КОМПОНЕНТ

N

1

ПРОЦЕСОР

Е

1

1

1

1

1

ТВЪРД ДИСК

LAN Карта

RAM Памет

ДРУГИ

МОНИТОР

Е

Е

Е

Е

Е

На фигура 3 е илюстриран и видът на еднотипните връзки, а именно – N към 1. Това е така, защото entity типоветена конкретните компонентиса само номенклатури т.е. в тях се изброява какъв е компонента. Останалата информация за конкретни дати и брой компоненти се предоставя от обобщаващиятentity тип *компонент*. Линията между обобщения компонент и конкретния е двойна, защото така се изразява структурното ограничение total participation, което означава, че всяка инстанция от entity процесор, например, трябва да участва във връзката „е“.

Обобщено представяне на смисловите връзки (relationship types)между отделните entity типове от предметната област:

* Всеки компютър е притежаван от точно един служител.
* Всеки служител може да има повече от един компютър.
* Всеки служител принадлежи към отдел.
* Във един отдел има много служители.
* Всеки служител има определена длъжност.
* Една длъжност може да бъде дадена на много служители.
* Всеки компонент е част от точно един компютър.
* Един компютър може да съдържа повече от един компонент.
* Един компютър има точно един тип (десктоп, лаптоп и т.н.).
* Един тип може да се даде на много компютри.
* Всеки един компонент има точно един тип (процесор, LAN и т.н.).
* Един тип може да се даде на много компоненти.
* Всеки компютър се помещава в точно един офис.
* Един офис може да помещава много компютри.
* Всеки офис се намира в точно един град.
* Един град може да има много офиси.
* Един град може да е част от точно една община.
* Една община съдържа множество градове.
* Една община се съдържа в точно една област.
* Една област съдържа множество общини.
* Един компютър има точно един производител.
* Един производител може да е произвел множество компютри.
* Един принтер се намира в точно един офис.
* Един офис може да има множество принтери.
* Един комутатор се намира в точно един офис.
* Един офис може да има множество комутатори.
* Едно UPS устройство може да бъде в точно един офис.
* Един офис може да съдържа множество UPS устройства.
* Един маршрутизатор може да се помещава в точно един офис.
* Един офис може да съдържа множество комутатори.
* Една дейност може да има точно един тип.
* Един тип дейност може да се даде на множество записи.

**2.1.3. Обобщение на типове същности и връзки в таблица**

В табл. 1 са посочени всички възможни връзки между entity типовете от предметната област на приложението за системна администрация и инвентаризация на ИТ оборудване.

|  |  |  |
| --- | --- | --- |
| Entity | Relationship | Entity |
| Компютър | Съдържа | Компонент |
| Компонент | Може да бъде | Процесор |
| Компонент | Може да бъде | Графична карта |
| Компонент | Може да бъде | Оптично устройство |
| Компонент | Може да бъде | Твърд диск |
| Компонент | Може да бъде | LAN карта |
| Компонент | Може да бъде | RAM памет |
| Компонент | Може да бъде | Монитор |
| Компонент | Може да бъде | Друг |
| Служител | Притежава | Компютър |
| Служител | Заема | Длъжност |
| Отдел | Съдържа | Служители |
| Компютър | Има | Тип |
| Компонент | Има | Тип |
| Дневни (запис) | Има | Тип |
| Област | Съдържа | Община |
| Община | Съдържа | Град |
| Град | Съдържа | Офис |
| Офис | Помещава | Компютър |
| Офис | Помещава | Принтер |
| Офис | Помещава | UPS устройство |
| Офис | Помещава | Комутатор |
| Офис | Помещава | Маршрутизатор |
| Производител | Произвежда | Компютър |
| Производител | Произвежда | Компонент |
| Производител | Произвежда | Принтер |
| Производител | Произвежда | Комутатор |
| Производител | Произвежда | UPS устройство |
| Производител | Произвежда | Маршрутизатор |

**Таблица 1**

**2.1.4. Обобщен ER модел на предметната област**

N

1

N

1

ГРАД

СЪДЪРЖА

ОБЩИНА

СЪДЪРЖА

ОБЛАСТ

ПРИНТЕР

N

1

1

ПОМЕЩАВА

1

ОФИС

N

СЪДЪРЖА

N

КОМПЮТЪР

ПРОИЗВЕЖДА

N

ПОМЕЩАВА

1

ПРИНАДЛЕЖИ НА

1

N

1

ПРОИЗВЕЖДА

1

N

ПРОИЗВОДИТЕЛ

**Фигура 4**

1

ВИДЕО КАРТА

Е

1

ОПТИЧНО У-ВО

Е

N

N

N

N

N

N

N

КОМПОНЕНТ

N

1

ПРОЦЕСОР

Е

1

1

1

1

1

ТВЪРД ДИСК

LAN Карта

RAM Памет

ДРУГИ

МОНИТОР

Е

Е

Е

Е

Е

ДЛЪЖНОСТ

ОТДЕЛ

1

1

N

N

СЛУЖИТЕЛ

ЗАЕМА

РАБОТИ В

N

СЪДЪРЖА

Фиг. 4 представлява обобщеният ER модел на предметната област на приложението, като не са описани всички връзки, защото са еднакви и имат еднакъв смисъл. Този ER модел напълно еднозначно и точно определя връзките между entity типовете за конкретното програмно осигуряване.

**2.2. Описание на релационният модел на базата данни**

Релационният модел на базата данни е резултат от етапа на логическия дизайн в процеса на проектиране на една информационна управляваща система.На този етап се осъществява връзката между концептуалният модел и релационният модел, който пък, от своя страна, е зависим от конкретната машина за бази данни. За да се извърши преход от концептуален модел (ER модел) към релационен модел на данните, се следват определени процедури, които следва да бъдат споменати накратко:

* Нормалните entity типове (strong entity types)се превръщат в имена на таблици, чиито колони представляват всички прости атрибути (съставните участват само с декомпозираните си), като се избира първичен ключ от кандидат-ключовете.
* Слабите entity типове (ако има такива) участват в таблицата чрез простите си атрибути, като включват и външен ключ към типа собственик – той и останалите атрибути оформят частичния ключ (partial key)на слабият тип същност.
* Конвертирането на връзка от тип 1:1 става по следните три начина:
  + Подход „външен ключ“ (foreign key approach) – избира се единият entity тип (обикновено този с пълно участие (total participation)) и в неговата таблица се добавя външен ключ към другата таблица. Това е най-често прилаганият подход.
  + Подход „комбинирани таблици“ (merged relation approach) – двата entity типа се сливат в една таблица. Възможен е само когато и двата типа са с пълно участие във връзката.
  + Подход „връзка в таблица“ (cross-reference approach/relationship relation)–крайно неефективен за този тип връзка. На практика се прилага само при връзки от тип M:N, където няма друг вариант.
* Конвертиране на връзка от тип 1:N – в таблицата от страна N на връзката се добавя външен ключ към таблицата от страна 1 на връзката.
* Превръщане на връзка M:N – става чрез отделна таблица, която съдържа два външни ключа към таблиците от двете страни на връзката, както и идентификатор на всяка една връзка в таблицата (ID). Тук се намират и всички атрибути на самата връзка.
* Превръщане на многостойностен атрибут (multivalued attribute) – създава се нова таблица, която съдържа поредната стойност на атрибута и външен ключ към таблицата, за която се отнася атрибута.
* Конвертиране на n-арна връзка – създава се нова таблица, съдържаща толкова на брой външни ключове, колкото entity типове участват във връзката. Участват и атрибутите на връзката.

Освен тези правила за превръщане на ER модела в релационен модел, съществуват и определени процедури, които дават представа за това, колко добър и ефективен е дизайна на релационния модел. Това са т.нар. нормални форми на базата данни. На практика, те са три, но в теорията съществуват още две. Описание на трите нормални форми:

* Първа НФ – данните във всички колони (атрибути) трябва да са атомични (неделими) т.е. да няма релация (таблица) в релация или многостойностен атрибут, който е вграден в самата таблица.
* Втора НФ – отнася се само за първични ключове, включващи повече от един атрибут. Не трябва да има неключови атрибути в таблицата, които да са функционално зависими от първичния ключ.
* Трета НФ (Boyce-Codd NF)–не трябва да има транзитивна зависимост между първичен ключ и неключов атрибут т.е. не трябва да има неключови атрибути, които са функционално определени от други неключови атрибути.

В приложението за системна администрация и инвентаризация на ИТ оборудване са спазени и трите нормални форми на нормализиране на базата данни, както ще стане ясно в следващите секции.

**2.2.1. Структура на релационните таблици**

|  |  |  |
| --- | --- | --- |
| **Таблица VEHICLE** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **PRODUCTION\_YEAR** | DATE | Тип на дейността (външен ключ) |
| **OWNER\_ID** | INTEGER | Дата на дейността |
| **DESCR** | VARCHAR(100) | Бележки към дейността |

Тази таблица съдържа данни за всички дейности, които се извършват в ИТ фирмата. Тя служи и като дневник, и като календар.

Тя кореспондира с Entity „Дневник (запис)“ от ER модела на базата данни.

Таблицата ACTIVITY\_RECORDS\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица ACTIVITY\_TYPE\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **ACTIVITY** | VARCHAR (20) | Дейност |

Тази таблица съдържа данни за всички типове дейности, които се извършват в ИТ фирмата. Тя служи като тяхна номенклатура.

Тя кореспондира с Entity „Тип“ (на дейност) от ER модела на базата данни.

Таблицата ACTIVITY\_TYPE\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица CITY\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **EKATTE** | INTEGER | Уникален за нас. място код |
| **NAME** | VARCHAR(20) | Име на града |
| **MUNICIPALITY\_ID** | INTEGER | Община (външен ключ) |

Тази таблица съдържа данни за градовете, в които могат да се намират офисите.

Тя кореспондира с Entity „Град“ от ER модела на базата данни.

Таблицата CITY\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица COMPONENT\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **COMPUTER\_ID** | INTEGER | Компютър (външен ключ) |
| **COMPONENT\_TYPE\_ID** | INTEGER | Тип на компонент (външен ключ) |
| **COMPONENT\_ID** | INTEGER | Конкретен компонент (външен ключ) |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на фактура |
| **SERIAL\_NR** | VARCHAR(40) | Сериен номер |

Тази таблица съдържа данни за компонентите на даден компютър.

Тя кореспондира с Entity „Компонент“ от ER модела на базата данни.

Таблицата COMPONENT\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица COMPUTER\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **COMPUTER\_TYPE\_ID** | INTEGER | Тип на компютър (външен ключ) |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **INTERFACES\_DESCR** | VARCHAR (100) | Описание на интерфейси |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на фактура |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **SCRAP\_DATE** | DATE | Дата на бракуване |
| **OFFICE\_ID** | INTEGER | Офис (външен ключ) |
| **PERSON\_ID** | INTEGER | Собственик/служител (външен ключ) |

Тази таблица съдържа данни за компютрите в ИТ фирмата.

Тя кореспондира с Entity „Компютър“ от ER модела на базата данни.

Таблицата COMPUTER\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица COMPONENT\_TYPE\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **COMPONENT\_TYPE** | VARCHAR(20) | Тип на компонент |

Тази таблица съдържа данни за типовете компоненти, които даден компютър може да съдържа.

Тя кореспондира с Entity „Тип“ (на компонент) от ER модела на базата данни.

Таблицата COMPONENT\_TYPE\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица COMPUTER\_TYPE\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **COMP\_TYPE** | VARCHAR(10) | Тип на компютър |

Тази таблица съдържа данни за типовете на компютрите в ИТ фирмата.

Тя кореспондира с Entity „Тип“ (на компютър) от ER модела на базата данни.

Таблицата COMPUTER\_TYPE\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица CPU\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **CLOCK\_MHZ** | INTEGER | Тактова честота |
| **CORE\_NUM** | INTEGER | Брой ядра |
| **CACHE\_KB** | INTEGER | Кеш памет |

Тази таблица съдържа данни за процесорите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Процесор“ от ER модела на базата данни.

Таблицата CPU\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица DEPARTMENT\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **NAME** | VARCHAR(20) | Отдел |

Тази таблица съдържа данни заотделите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Отдел“ от ER модела на базата данни.

Таблицата DEPARTMENT\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица HDD\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **FORMFCTR** | FLOAT | Форм фактор |
| **CAPACITY\_GB** | INTEGER | Капацитет |
| **CACHE\_KB** | INTEGER | Кеш памет |
| **RPM** | INTEGER | Обороти/Минута |

Тази таблица съдържа данни затвърдите дискове в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Твърд диск“ от ER модела на базата данни.

Таблицата HDD\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица LAN\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **SPEED\_GBPITS** | INTEGER | Скорост (Гбит/с) |

Тази таблица съдържа данни за LAN картите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „LAN карта“ от ER модела на базата данни.

Таблицата LAN\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица MANUFACTURER\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **NAME** | VARCHAR(50) | Име на производител |

Тази таблица съдържа данни запроизводителите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Производител“ от ER модела на базата данни.

Таблицата MANUFACTUER\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица MONITOR\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **SCREEN\_SIZE** | FLOAT | Размер на екрана |

Тази таблица съдържа данни замониторите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Монитор“ от ER модела на базата данни.

Таблицата MONITOR\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица MUNICIPALITY\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **NAME** | VARCHAR(20) | Име на община |
| **REGION\_ID** | INTEGER | Област (външен ключ) |

Тази таблица съдържа данни заобщините в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Община“ от ER модела на базата данни.

Таблицата MUNICIPALITY\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица OFFICE\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **CITY\_ID** | INTEGER | Град (външен ключ) |
| **NAME** | VARCHAR(50) | Име на офис |
| **ADDRESS** | VARCHAR(100) | Адрес |

Тази таблица съдържа данни заофисите в базата данни.

Тя кореспондира с Entity „Офис“ от ER модела на базата данни.

Таблицата OFFICE\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица OPTICAL\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **WRITESPEED** | INTEGER | Скорост на запис |
| **READSPEED** | INTEGER | Скорост на четене |
| **FORMATS** | VARCHAR(50) | Разпознавани формати |

Тази таблица съдържа данни заоптичните устройства в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Оптично устройство“ от ER модела на базата данни.

Таблицата OPTICAL\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица OTHER\_COMP\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **DESCR** | VARCHAR(200) | Описание |

Тази таблица съдържа данни закомпонентите от категория „Други“ в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Други“ от ER модела на базата данни.

Таблицата OTHER\_COMP\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица PERSON\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **EGN** | VARCHAR(20) | Егн на служител |
| **FNAME** | VARCHAR(20) | Име |
| **SNAME** | VARCHAR(20) | Презиме |
| **LNAME** | VARCHAR(20) | Фамилия |
| **PHONE** | VARCHAR(20) | Телефон |
| **EMAIL** | VARCHAR(50) | Е-поща |
| **POSITION\_ID** | INTEGER | Длъжност (външен ключ) |
| **DEPARTMENT\_ID** | INTEGER | Отдел (външен ключ) |

Тази таблица съдържа данни заслужителите във ИТ фирмата.

Тя кореспондира с Entity „Служител“ от ER модела на базата данни.

Таблицата PERSON\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица POSITION\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **NAME** | VARCHAR(20) | Име на длъжност |

Тази таблица съдържа данни задлъжностите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Длъжност“ от ER модела на базата данни.

Таблицата POSITION\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица PRINTER\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **PAGES\_PER\_MIN** | INTEGER | Страници/мин. |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на факт. |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **SCRAP\_DATE** | DATE | Дата на брак. |
| **OFFICE\_ID** | INTEGER | Офис (външен ключ) |

Тази таблица съдържа данни запринтерите в ИТ фирмата.

Тя кореспондира с Entity „Принтер“ от ER модела на базата данни.

Таблицата PRINTER\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица RAM\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **CAPACITY\_MB** | INTEGER | Капацитет |
| **CLOCK\_MHZ** | INTEGER | Работна честота |

Тази таблица съдържа данни за RAM паметите в базата данни и служи като номенклатура.

Тя кореспондира с Entity „RAM памет“ от ER модела на базата данни.

Таблицата RAM\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица REGION\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **NAME** | VARCHAR(20) | Име на област |

Тази таблица съдържа данни заобластите на населените места в базата данни и служи като номенклатура.

Тя кореспондира с Entity „Област“ от ER модела на базата данни.

Таблицата REGION\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица ROUTER\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **MAX\_STATIONS** | INTEGER | Макс. станции |
| **RANGE\_METERS** | INTEGER | Обхват (м) |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на факт. |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **SCRAP\_DATE** | DATE | Дата на брак. |
| **OFFICE\_ID** | INTEGER | Офис (външен ключ) |

Тази таблица съдържа данни замаршрутизаторите в ИТ фирмата.

Тя кореспондира с Entity „Маршрутизатор“ от ER модела на базата данни.

Таблицата ROUTER\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица SWITCH\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **SPEED\_GBIT** | INTEGER | Скорост (Гбит/с) |
| **PORT\_NUM** | INTEGER | Брой портове |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на факт. |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **SCRAP\_DATE** | DATE | Дата на брак. |
| **OFFICE\_ID** | INTEGER | Офис (външен ключ) |

Тази таблица съдържа данни закомутаторите в ИТ фирмата.

Тя кореспондира с Entity „Комутатор“ от ER модела на базата данни.

Таблицата SWITCH\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица UPS\_TBL** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **MAX\_STATIONS** | INTEGER | Макс. станции |
| **MAX\_MINS** | INTEGER | Макс. минути |
| **INVOICE\_DATE** | DATE | Дата на фактура |
| **INVOICE\_NR** | INTEGER | Номер на факт. |
| **EXPIRE\_DATE** | DATE | Дата на гаранция |
| **SCRAP\_DATE** | DATE | Дата на брак. |
| **OFFICE\_ID** | INTEGER | Офис (външен ключ) |

Тази таблица съдържа данни за UPS устройствата в ИТ фирмата.

Тя кореспондира с Entity „UPS устройство“ от ER модела на базата данни.

Таблицата UPS\_TBL е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

|  |  |  |
| --- | --- | --- |
| **Таблица VGA\_TBLN** | | |
| **Атрибути** | **Тип** | **Описание** |
| **ID** | INTEGER | Първичен ключ |
| **MAN\_ID** | INTEGER | Производител (външен ключ) |
| **MODEL** | VARCHAR(50) | Модел |
| **CAPACITY\_MB** | INTEGER | Капацитет |
| **CPUCLOCK\_MHZ** | INTEGER | Работна честота |
| **CORE\_NUM** | INTEGER | Брой ядра |

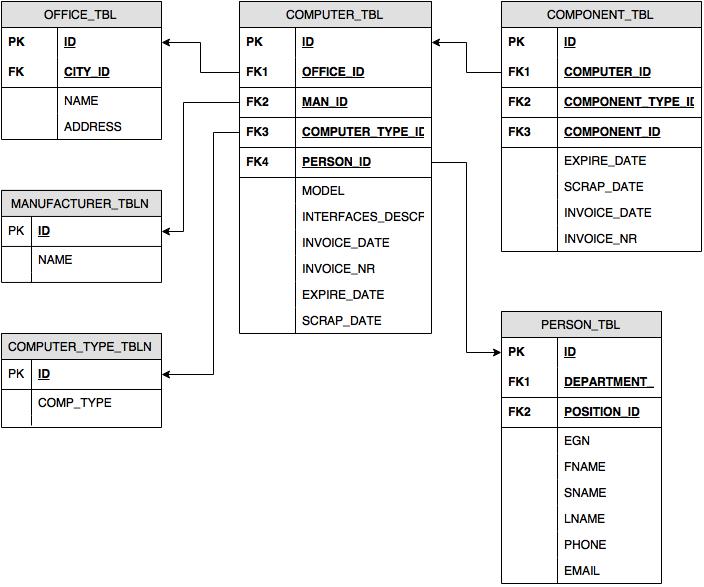
Тази таблица съдържа данни заграфичните адаптери в базата данни и служи като номенклатура.

Тя кореспондира с Entity „VGA устройство“ от ER модела на базата данни.

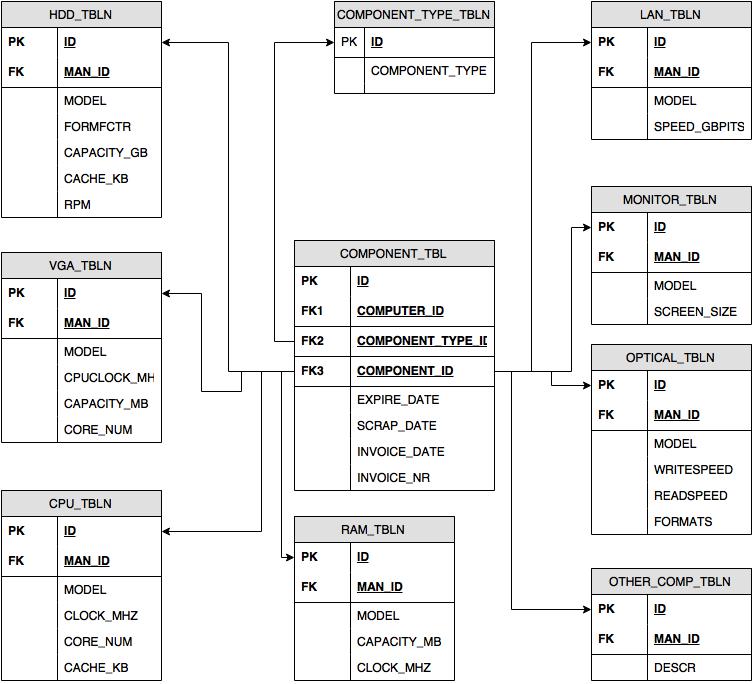
Таблицата VGA\_TBLN е приведена и в трите нормални форми, защото:

* Всички атрибути са прости/атомични т.е. няма съставни или многостойностни атрибути.
* Избран е само един атрибут за първичен ключ (ID), а не множество атрибути, което автоматично означава, че базата данни е приведена във втора нормална форма.
* Няма транзитивна зависимост между първичният ключ и неключов атрибут.

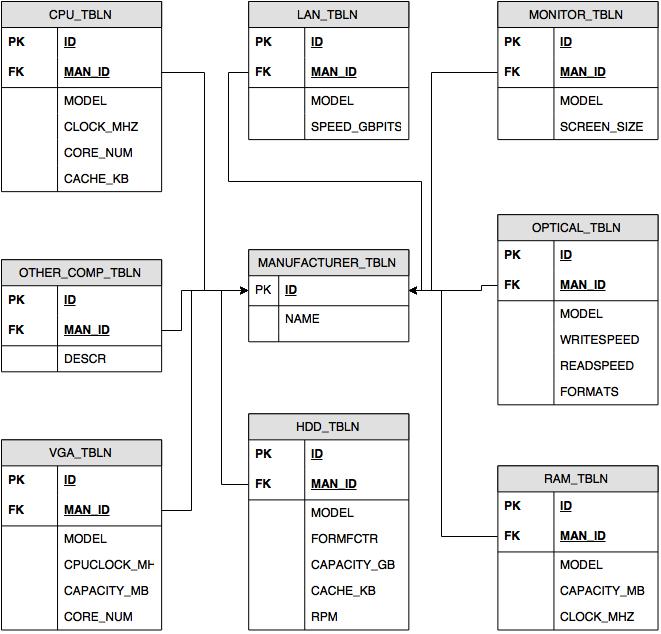
**2.2.2. Релационни таблици и връзките между тях**



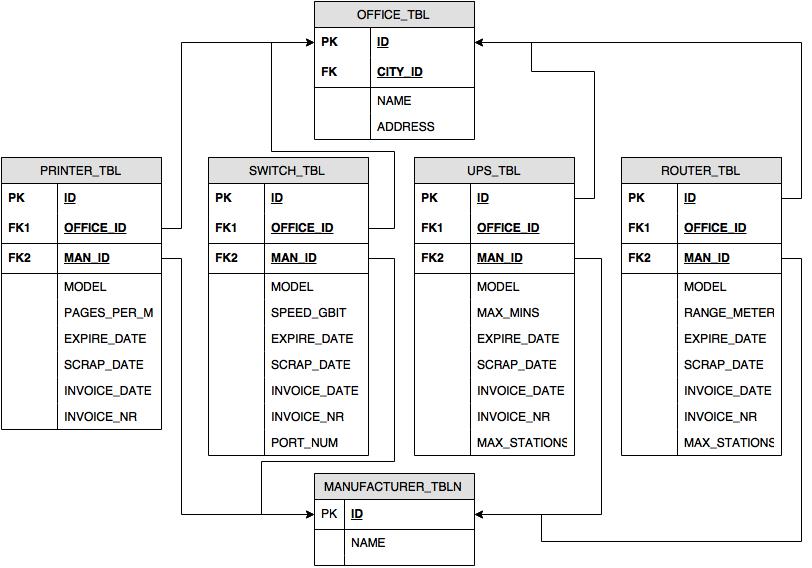
**Фигура 5: Връзки на таблица „Компютър“ с таблиците „Компонент“, „Служител“, „Тип на компютър“, „Производител“ и „Офис“**



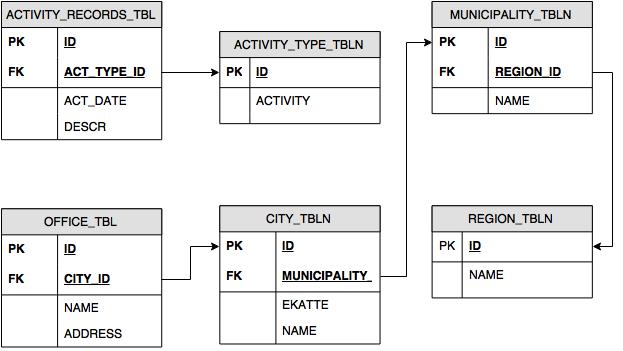
**Фигура 6: Връзки на таблица „Компонент“ с таблиците „Тип на компонент“, „LAN карта“, „Монитор“, „Оптично устройство“, „Други“, „RAM памет“, „Процесор“, „VGA устройство“ и „Твърд диск“**



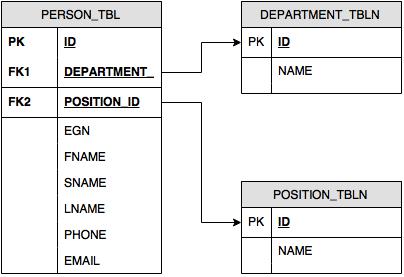
**Фигура 7: Връзки на таблица „Производител“ с таблиците „Процесор“, „Твърд диск“, „Други“, „Оптично устройство“, „VGA устройство“, „RAM памет“, „LAN карта“ и „Монитор“**



**Фигура 8: Връзки на таблиците „Офис“ и „Производител“ с таблиците „Комутатор“, „Маршрутизатор“, „Принтер“ и „UPS устройство“**



**Фигура 9: Връзка на таблиците „Дневник (запис)“ и „Тип на дейност“. Връзки на таблиците „Офис“, „Град“, „Община“, и „Област“**



**Фигура 10: Връзки на таблица „Служител“ с таблиците „Отдел“ и „Длъжност“**

**Глава 3**

**Обосновка за избор на машина за база данни, език за програмиране и програмна среда**

**3.1. Обосновка за избор на машина за бази данни**

Изборът на тип система за управление на бази данни е релационната машина за бази данни (RDBMS – relational database management system). Основните характеристики на този тип машини за бази данни са мощност, която се изразява в свойството scalability (мащабируемост), както и надеждност.

Като машина за бази данни е избран InterBase Server DBMSv6.5, защото притежава едни от най-важните характеристики на една система за управление на база данни – ефективност и бързодействие.Други характерни черти на InterBase са: висока производителност, пълен набор от функции, ниска цена, машина за бази данни с минимални системни изисквания. Работи под платформите Windows, Linux, Mac.Вградени са средства за сигурност като потребителски акаунти и различни права на достъп, и криптиране на данните. Машината за бази данни може да работи локално или на отдалечен сървър. Поддържа стандарта SQL 92 и мениджмънт на трансакциите.

Може да се направи сравнение по областта, в която се възнамерява да се ползва системата за управление на база данни, а именно – ако изберем една система на Oracle за база данни за нашата задача, тогава цената е крайно висока и не е ефективно изразходването на толкова голям финансов ресурс. Но ако изберем PostgreSQL, тогава сме ограничени от платформите на приложимост. InterBase е безплатен (или с минимална цена) и предлага много добри възможности за интеграция в C++ Builder.

**3.2. Обосновка за избор на език за програмиране**

За изготвянето на тази дипломна работа е избран езикът С++. Наборът от възможности, които предлага на програмиста е много голям, като едни от основните предимства са обектно-ориентиран стил на програмиране и висока ефективност на изходният код (за разлика от Java, тук не се използва виртуална машина). ООП дава възможност за:

* Потребителски дефинирани типове (класове) – по този начин програмистът винаги може да моделира данни, с които ще работи. Също така, един код, който е написан веднъж, може да се използва в бъдещи проекти. Във този ред на мисли, следва и да се спомене огромната библиотека от готови функции и класове, които езикът предлага и които могат да се доразработват, благодарение на механизма на наследяване.
* Полиморфизъм – с всеки един новосъздаден клас програмистът може да придава определен смисъл на вече познати оператори в контекста на новия клас. По този начин операторите стават Overloaded. Полиморфизмът представлява много форми на едно и също нещо (оператор, функция).
* inline функции – ако компилаторът го позволява, тогава той ще включи директно функцията в кода вместо да прави специално извикване (което води до латентност).
* Класове-шаблони – това дава възможност поведението на един нов клас да бъде дефинирано, като самите типове данни стават променливи от класа
* Виртуални функции – дават възможността да се дефинира функции с едно и също име и една и съща сигнатура в даден клас и неговия производен. Но при извикване ще се извика функцията, която съответства на класа.

**3.3. Обосновка за избор на програмна среда**

Borland C++ Builder (BCB)е среда за бърза разработка на приложенията (RAD IDE – rapid application development), която предлага богат набор от компоненти и функционалности. Ето и някои от тях:

* Наличие на библиотеката VCL (visual component library) – тя предлага множество компоненти, които са разработени и могат да бъдат наследявани и разширявани от програмиста
* Компонентно-ориентирано програмиране – към множеството възможности на ООП са добавени и събитията, които внасят голямата разлика в стила на програмиране (програмиста може да прави определени неща при определени събития). Една компонента „затваря“ в една капсула класът и събитията към него.
* Стилът на програмиране е contract-free (за разлика от MS MFC) – не се изискват някакви условия, за да функционира съответният код

**Глава 4**

**Описание на основните програмни модули и на взаимодействието между тях**

**4.1. Описание на форма модул за данни DModFRM**

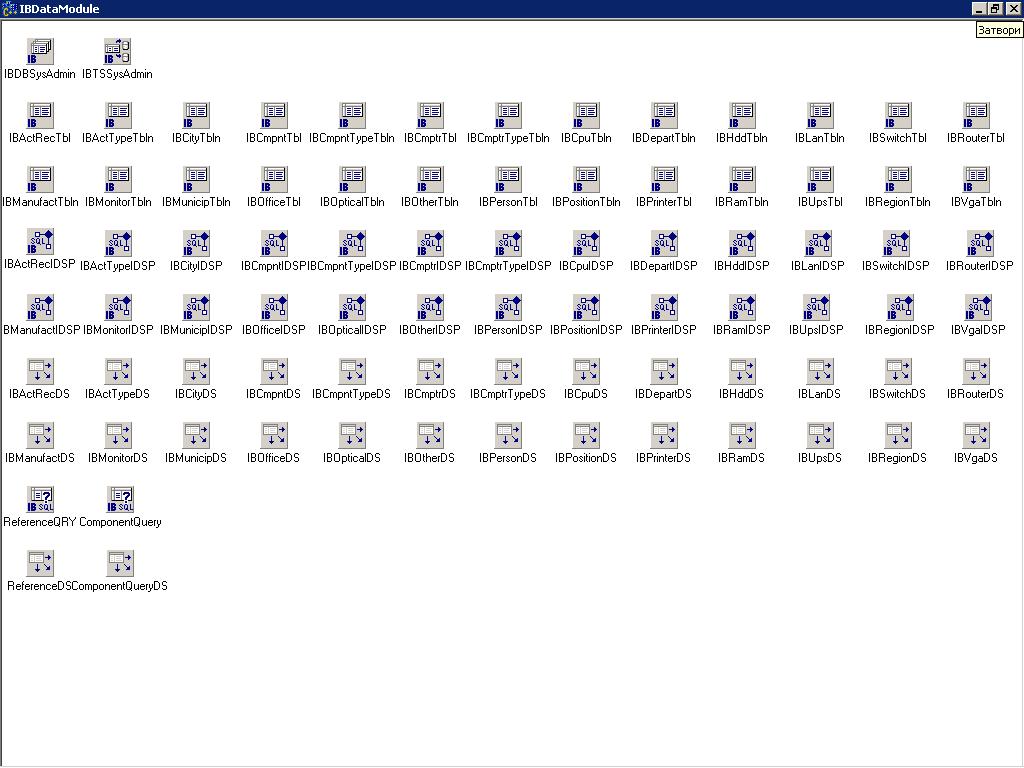
Приложението за системна администрация и инвентаризация на ИТ оборудване е създадено чрез интегрираната среда за разработка (IDE) Borland C++ Builder. Поради тази причина, това програмно осигуряване използва концепциите и методите, които средата имплементира.

Основен способ за връзка с база данни и работа с нея е т.нар модул за данни (Data Module). Той е предпочитан и използван, защото предлага следните предимства:

* Лесно управление на визуалните компоненти
* Разделение на визуални (за потребителя) и невизуални (за осъществяване на функционалността от гл. точка на базата данни) компоненти
* Способност за споделяне и използване на един и същ модул за данни между различни приложения – така връзката с базата данни се унифицира
* Наследяване на модула за данни
* Капсулиране на правилата за референциален интегритет и валидация на данните извън базата данни

**4.1.1. Цели логика на DModFRM**

Целта на модула за данни DModFRM е да осъществи връзката с базата данни, като тава става с определен компонент, а именно - компонент за връзка с базата данни (TIBDatabase). Той, от своя страна, изисква компонент за трансакциите към базата данни (TIBTransaction). За да може извлечените данни да достигнат до визуалната част на приложението, трябват и компоненти, които обработват и съхраняват данните от всяка една таблица по отделно (TIBTable). Таблиците изискват компонент за осъществяване на достъп до данните (TDataSource). След като вече достъпът е осигурен от TDataSource компонент, остава данните да бъдат визуализирани от компонента TDBGrid. Ето как изглежда и самият модул за данни:



**Фигура 11: Форма DModFRM**

**4.1.2. Съставни части на DModFRM**

Модулът за данни към приложението за системна администрация и инвентаризация на ИТ оборудване се състои от следните компоненти:

* IBDBSysAdmin – компонент за връзка с бази данни
* IBTSSysAdmin – компонент за трансакции към базата данни
* Компоненти за извличане на данни от конкретна таблица – IBActRecTbl, IBActTypeTbln, IBCityTbln, IBCmpntTbl, IBCmpntTypeTbln, IBCmptrTbl, IBCmptrTypeTbln, IBCpuTbln, IBDepartTbln, IBHddTbln, IBLanTbln, IBSwitchTbl, IBRouterTbl, IBManufactTbln, IBMonitorTbln, IBMunicipTbln, IBOfficeTbl, IBOpticalTbln, IBOtherTbln, IBPersonTbl, IBPositionTbln, IBPrinterTbl, IBRamTbln, IBUpsTbl, IBRegionTbln и IBVgaTbln.
* Компоненти за достъпване на данни от конкретна таблица – IBActRecDS, IBActTypeDS, IBCityDS, IBCmpntDS, IBCmpntTypeDS, IBCmptrDS, IBCmptrTypeDS, IBCpuDS, IBDepartDS, IBHddDS, IBLanDS, IBSwitchDS, IBRouterDS, IBManufactDS, IBMonitorDS, IBMunicipDS, IBOfficeDS, IBOpticalDS, IBOtherDS, IBPersonDS, IBPositionDS, IBPrinterDS, IBRamDS, IBUpsDS, IBRegionDS и IBVgaDS.
* Компоненти, които са процедури към базата данни (TIBStoredProc)и служат за извличане и увеличаване на стойността на колоната ID на съответната таблица с цел автоматично инкрементиране – IBActRecIDSP, IBActTypeIDSP, IBCityIDSP, IBCmpntIDSP, IBCmpntTypeIDSP, IBCmptrIDSP, IBCmptrTypeIDSP, IBCpuIDSP, IBDepartIDSP, IBHddIDSP, IBLanIDSP, IBSwitchIDSP, IBRouterIDSP, IBManufactIDSP, IBMonitorIDSP, IBMunicipIDSP, IBOfficeIDSP, IBOpticalIDSP, IBOtherIDSP, IBPersonIDSP, IBPositionIDSP, IBPrinterIDSP, IBRamIDSP, IBUpsIDSP, IBRegionIDSP и IBVgaIDSP.
* Два компонента от тип заявка (TIBQuery), които служат за изпълнение на заявките по извличане на информация за компютър и извършване на определени заявки – CompnentQRY и ReferenceQuery.
* Два компонента за достъпване на информацията (TDataSource), която се връща от двете заявки към базата данни – ComponentQueryDS и ReferenceDS.

**4.1.3. Събития на DModFRM**

В модула за данни са реализирани определени събития, които извършват определени процедури. Те са:

* Събитие **BeforePost()**на всяка таблица от модула за данни. То настъпва точно преди данните да бъдат записани в съответната таблица, като автоматично увеличава и извлича стойността на полето ID за записа, който предстои да бъде записан.

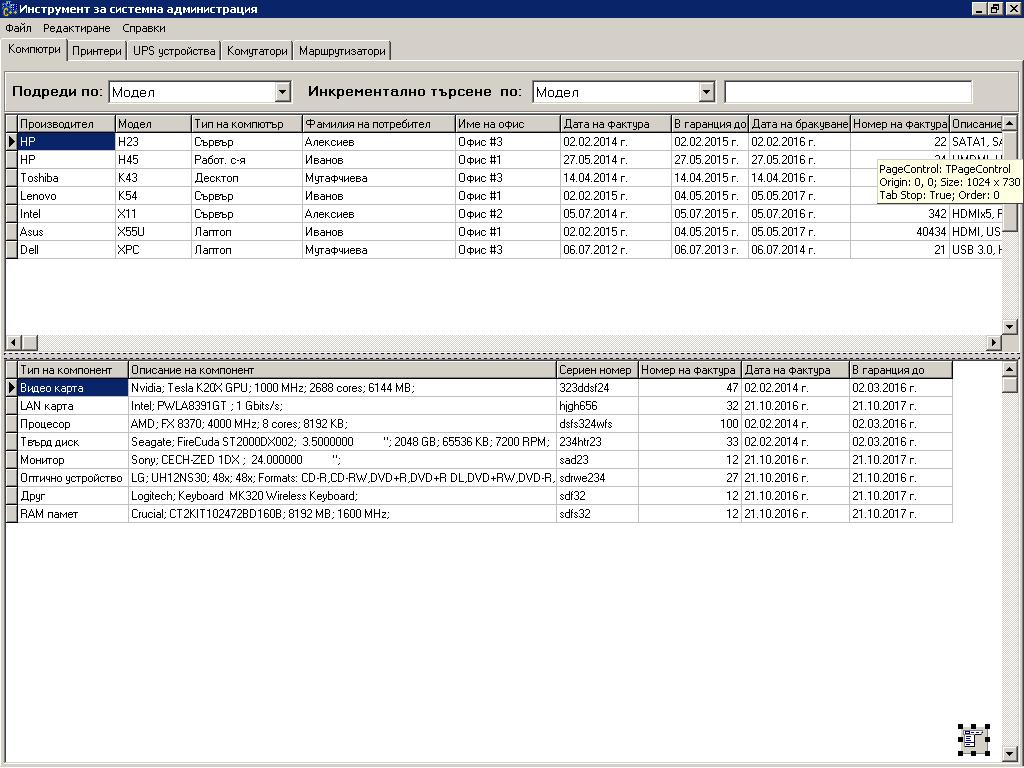
**4.2.Описание на главна форма IBSysAdminMainFRM**

Главната форма на едно приложение е тази форма, която потребителят вижда, когато се стартира приложението. Обикновено тя служи за управление и извикване на други помощни форми, като и самата тя най-често съдържа определени функционалности.

В програмата за системна администрация и инвентаризация на ИТ оборудване главната форма IBSysAdminMainFRM съдържа значителна част от функционалността на приложението.Формата представлява графичният потребителски интерфейс на приложението, като съдържа в себе си стандартни елементи и работи на базата на общи принципи.

**4.2.1. Цел и логика на главна форма IBSysAdminMainFRM**

Целта на главната форма IBSysAdminMainFRM е да изведе и визуализира наличната информация от базата данни за всички устройства, а именно – компютри и техните компоненти, принтери, комутатори, маршрутизатори и UPS устройства. Също така, горепосочените устройства могат да се редактират, добавят и изтриват в зависимост желанието на потребителя. Главната форма е организирана в раздели (табове), като на всеки раздел съответства определен вид устройство. IBSysAdminMainFRM съдържа и главно меню, което служи за навигация в приложението. То предлага възможности за превключване към определен раздел (като алтернатива на кликането върху таба), добавяне и изтриване на компютър, добавяне и изтриване на компонент, редактиране на номенклатурите в базата данни, извършване на справки и т.н. Стандартният интерфейс позволява лесна навигация и интуитивен контрол върху системата. Ето как изглежда главната форма на приложението за системна администрация и инвентаризация на ИТ оборудване:



**Фигура 12: Форма IBSysAdminMainFRM**

**4.2.2. Съставни части на главна форма IBSysAdminMainFRM**

Формата се състои от следните компоненти:

* Компонент за главно меню MainMenu (TMainMenu) – това екомпонентът, който имплементира главното меню в приложението. В него се съдържат функциите по промяна на раздел от PageControl компонента, изтриване на компютър, добавяне и премахване на компонент, редактиране на номенклатури в базата данни, извършване на справки, редактиране на дневник, редактиране на служители и изход от приложението.
* Компонент PageControl (TPageControl) – този компонент реализира табовете на приложението, в които се съдържа информацията за съответното устройство
* Всеки един раздел съдържа по два компонента от клас TComboBox, които служат за указване начина на подреждане на визуализираните данни и критерия за търсене на данни. Работата им се базира на това, че променят текущият индекс на съответната таблица. Включен е и компонент от клас TEditBox за въвеждане и търсене на данни. Тези компоненти са: OrderComboCmptr, SearchComboCmptr, CmptrTblSearchEB, OrderComboSwitch, SearchComboSwitch, SwitchTblSearchEB, OrderComboUps, SearchComboUps, UpsTblSearchEB, OrderComboRouter, SearchComboRouter, RouterTblSearchEB, OrderComboPrinter, SearchComboPrinter, PrinterTblSearchEB.
* Всеки един раздел на приложението съдържа компонент от клас TDBGrid, който служи за визуализиране и редактиране на данните от съответната таблица (функцията за изтриване на данни е имплементирана в компонента MainMenu). Използват се следните DBGrid компоненти: IBCmptrTblDBGrid, IBCmpntTblDBGrid, IBPrinterTblDBGrid, IBUpsTblDBGrid, IBSwitchTblDBGrid, IBRouterTblDBGrid.

**4.2.3. Събития на главна форма IBSysAdminMainFRM**

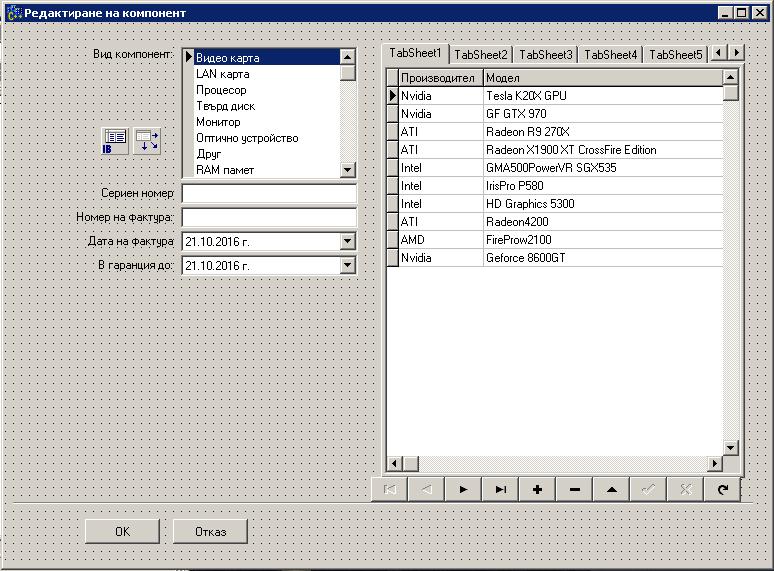
Обработват се следните събития:

* Събитие **OnExit ()**на всеки един DBGrid компонент – това събитие се обработва с цел записване на данните, които потребителят редактира, когато grid компонента загуби фокус. По този начин промените не се загубват.
* Събитие **OnChange ()** на всеки един EditBox компонент – настъпва, когато се промени съдържанието на полето за редактиране. Чрез това събитие се извършва т.нар инкрементално търсене т.е. търсене още в момента на въвеждане.
* Всички събития от тип **click**, които се случват в компонента за главно меню MainMenu. Тук се обработват следните типове събития: показване на друга форма, изтриване на компютър, редактиране на компонент и изход от програмата.
* Събитие **OnSelect ()**на ComboBox компонентите от различните табове – обработва се това събитие, вместо събитието OnClick () на съответния ComboBox, защото стилът на компонента csDropDownList, а това изисква събитие OnSelect (). Събитието настъпва, когато потребителят избере опция от падащото меню на ComboBox. Тогава, в зависимост от това дали ComboBox компонента е за смяна начина на показване на данните или критериий за търсене, се променя редът на показване на записите или критерия за търсене на запис.
* Събитие PageControl->**OnChanging ()**–настъпва точно преди да се промени раздела от PageControl компонента при клик върху друг таб. Резултатът от обработката на това събитие е изключване на опциите, които се предлагат от компонентът за главно меню MainMenu.
* Събитие PageControl->**OnChange ()** – настъпва веднага след като се смени разделът от компонент PageControl.Това събитие включва опции от менюто MainMenu, като те са съобразени с текущият таб и данните в него.

**4.3. Описание на форма за редактиране на компоненти ComponentEditFORM**

**4.3.1. Цел и логика на форма за редактиране на компоненти ComponentEditFORM**

Целта на тази форма е да предостави възможност на потребителя да добавя компоненти към даден компютър, както и да редактира съдържанието на номенклатурите с компоненти. Формата се стартира в модален режим чрез определена опция на главното меню. След това се избира тип на компонент, конкретен компонент и информация относно фактурата и гаранцията му. Накрая се потвърждава с бутон OK или се прави отказ посредством бутона Cancel. Ето как изглежда и самата форма:



**Фигура 13: Форма ComponentEditFORM**

**4.3.2. Съставни части на форма за редактиране на компоненти ComponentEditFORM**

Тази форма се състои от следните компоненти:

* Компоненти от клас DBGrid за визуализиране и редакция на номенклатурите с компютърни компоненти. Те са: IBVgaDBGrid, IBLanDBGrid, IBCpuDBGrid, IBMonitorDBGrid, IBOtherDBGrid, IBHddDBGrid, IBOpticalDBGrid, IBRamDBGrid
* Компоненти от клас TEditBox за въвеждане номер на фактура и сериен номер – CmpntInvNrEB и CmpntSerialNrEB
* Компоненти от клас TDateTimePicker за избор на дата на фактура и дата на гаранция – CmpntInvDateDTPicker и CmpntExpDateDTPicker
* Компонент от клас DBGrid за избор на тип на компонент –ComponentTypeDBGrid
* Компонент от клас TTable за връзка с таблицата с типове компоненти от базата данни – IBCmpntTypeTbln
* Компонент от клас DataSourceза достъпване на данните от таблицата с типове компоненти– IBCmpntTypeDS
* Компонент от клас TDBNavigator за редактиране на записите на номенклатурите с компоненти –ComponentDBNavigator
* Компонент от клас TPageControl за организиране на номенклатурите с компоненти в табове –ComponentPages
* Бутони от клас TBitBtn за потвърждение и отказ на промените – OKBitBtn и CancelBitBtn

**4.3.3. Събития на форма за редактиране на компоненти ComponenEditFORM**

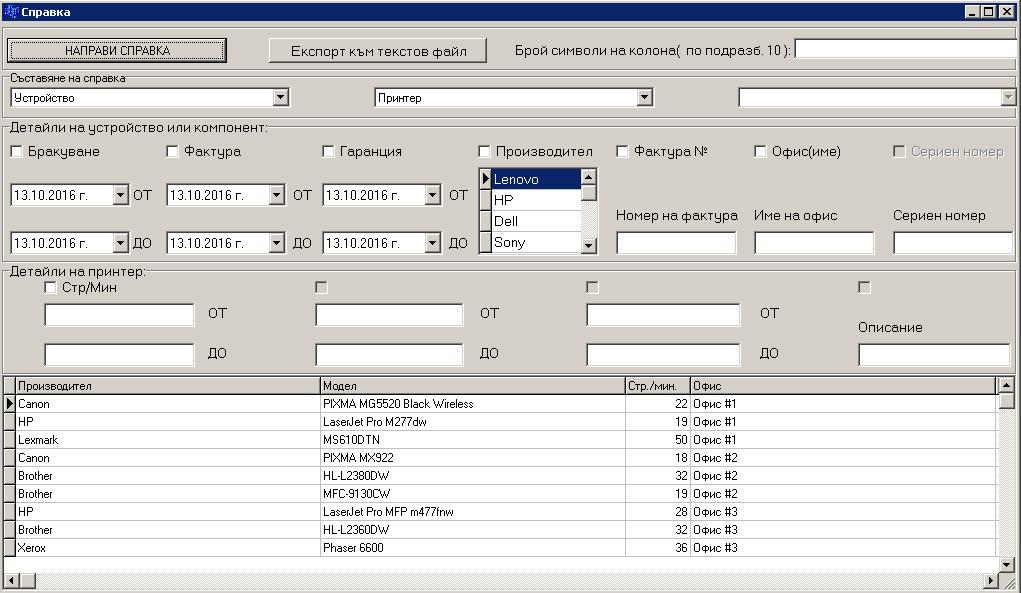
Формата съдържа следните събития:

* Събитие IBCmpntTypeTbln->**AfterSctroll ()** – настъпва, когато се промени текущият запис от grid компонента, съдържащ номенклатурата с типове компоненти. Обработва се, защото така се променя текущата активна страница на PageControl компонента и се показват данните от grid-a, съответстващ на таба. Също така се променя DataSource property компонента ComponentDBNavigator, за да може да се оперира с текущо избраната таблица.
* Събитие **FormCreate ()** – настъпва веднага след създаването на формата за редактиране на компонент. Обработва се, защото трябва да се визуализира само активната страница от PageControl компонента.
* Събитие **OnExit ()**на всички DBGrid компоненти, представляващи номенклатура с компютърни компоненти – настъпва при загуба на фокус от страна на съответния DBGrid компонент. Обработва се, за да може да се запазят промените, които потребителят е направил с даден запис, но е изрично запазен.

**4.4. Описание на форма за справки RefFORM**

**4.4.1. Цел и логика на форма за справки RefFORM**

Целта на формата за справки е да позволи на потребителите да съставят и отправят своите справки към базата данни чрез използване на графичния интерфейс вместо директно да се пишат на SQL. Това се осъществява като потребителят избира съдържанието на определени компоненти от клас ComboBox, клас TDateTimePicker и клас EditBox. След това се натиска и самият бутон, който изпраща заявката към машината за бази данни. Тази форма предлага и експорт, посредством отделен бутон, който записва информацията от направената справка в текстов файл. Целта е да се осигури възможност за принтиране на съдържанието от направената справка. Ето как изглежда и самата форма (за избрано устройство „принтер“, например):



**4.4.2. Съставни части на форма за справки RefFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране на резултатите от справката – ReferenceDBrid
* Два бутона от клас TBitBtn за изпращане на заявката и експортиране в текстов файл – ReferenceBitBtn и QuerySaveBitBtn
* Текстова кутия от клас TEditBox за въвеждане броя символи в колона при експортиране в текстов файл – QuerySaveEB
* Компоненти от клас TCheckBox служещи за отмятане на това, кои данни да се включат в заявката – PersonInfoCB\_1, PersonInfoCB\_2, PersonInfoCB\_3, ActivityCB\_1, ActivityCB\_2, ActivityCB\_3, CmpntDetCB\_1, CmpntDetCB\_2, CmpntDetCB\_3, CmpntDetCB\_4, CommonCB\_1, CommonCB\_2, CommonCB\_3, CommonCB\_4, CommonCB\_5, CommonCB\_6, CommonCB\_7
* Компоненти от клас TEditBox за въвеждане на специфичните параметри, на които ще отговарят резултатите от справката –PersonInfoEB\_1, PersonInfoEB\_2, PersonInfoEB\_3, OfficeEB\_1, OfficeEB\_2, ActivityEB, CmpntDetEB\_1, CmpntDetEB\_2, CmpntDetEB\_3, CmpntDetEB\_4, CmpntDetEB\_5, CmpntDetEB\_6, CmpntDetEB\_7, CommonEB\_1, CommonEB\_2, CommonEB\_3
* Компоненти от клас TComboBox за избиране на предварително зададени възможни параметри –MainCombo, DeviceCombo, ComponentCombo, ActivityCombo
* Компоненти от клас TDateTimePicker за определяне на дати и срокове към заявката –ActivityDTPicker\_1, ActivityDTPicker\_2, CommonDTPicker\_1, CommonDTPicker\_2, CommonDTPicker\_3, CommonDTPicker\_4, CommonDTPicker\_5, CommonDTPicker\_6

**4.4.3. Описание на събитията на форма за справка RefFORM**

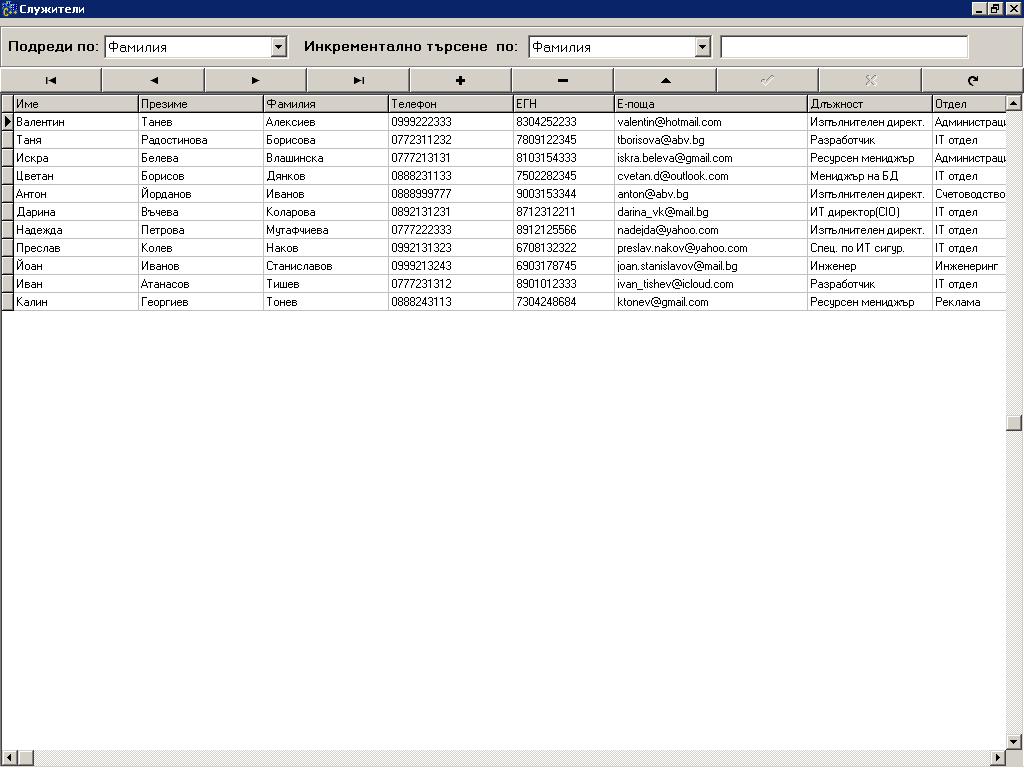
В тази форма се обработват следните събития:

* MainCombo->**OnSelect ()** – настъпва, когато се избере ред от компонента MainCombo. При това събитие се избира кои компоненти от формата да станат активни/видими, като така се улеснява потребителят, който съставя справката.
* DeviceCombo->**OnSelect ()** – настъпва, когато потребителят избере ред от комбинираната кутия за устройства. Тук отново се избира кои компоненти да останат активни/видими и кои не.
* ComponentCombo->**OnSelect ()** – настъпва, когато потребителят избере конкретен компонент (процесор, твърд диск и т.н.) от комбинираната кутия за компоненти. Събитието се обработва, защото така се определят кои елементи да останат активни/видими и какви да бъдат етикетите над текстовите кутии.
* ReferenceBitBtn->**OnClick ()**–настъпва, когато потребителят избере бутона за изпращане на справката. При това събитие се „сглобява“ самата SQL заявка, която остава невидима за потребителя. След това се настройва property-то SQL на компонента тип TQuery от модула за данни и се извиква методът му Open (), за да се изпълни заявката. Като последно действие, извършено при настъпването на това събитие, е изчистването на всички текстови кутии, премахването на отметките от checkbox елементите и връщане на началното състояние на комбинираните кутии.
* QuerySaveBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за запазване на справката в текстов файл. Действията, които се случват са: извикване на функцията Execute () на диалоговия компонент и предаване на резултата от него (FileName property-то)към моята функция за записване във файл – WriteReferenceToFile.
* RefFRM->OnClose () – настъпва преди да се затвори формата за справки. Проверява се дали е активна/отворена заявката от направената справка и ако да, тогава тя се деактивира.

**4.5. Описание на форма за редактиране на служители PersonFORM**

**4.5.1. Цел и логика на форма за редактиране на служители PersonFORM**

Целта на тази форма е да предостави възможност на потребителя да въвежда, изтрива, редактира данни за служителите в дадена ИТ фирма. Формата предлага избор на реда на записите при визуализация, както и търсене на записи по определен критерий. За изтриване на запис се ползва предоставеният DBNavigator контрол. Ето как изглежда формата:



**Фигура 15: Форма PersonFORM**

**4.5.2. Съставни части на форма за служители PersonFORM**

Формата се състои от следните компоненти:

* Два компонента от клас TComboBox за определяне реда на показване и критериите за търсене на запис - OrderComboPerson и SearchComboPerson
* Един компонент от клас TEditBox завъвеждане на данните, които ще се търсят –PersonTblSearchEB
* Компонент от клас TDBNavigator за изтриване, добавяне и редактиране на записи – PersonDBNavigator
* Компонент от клас TDBGrid за визуализиране и редактиране на записите за служителите –IBPersonTblDBGrid

**4.5.3. Събития на форма за служители PersonFORM**

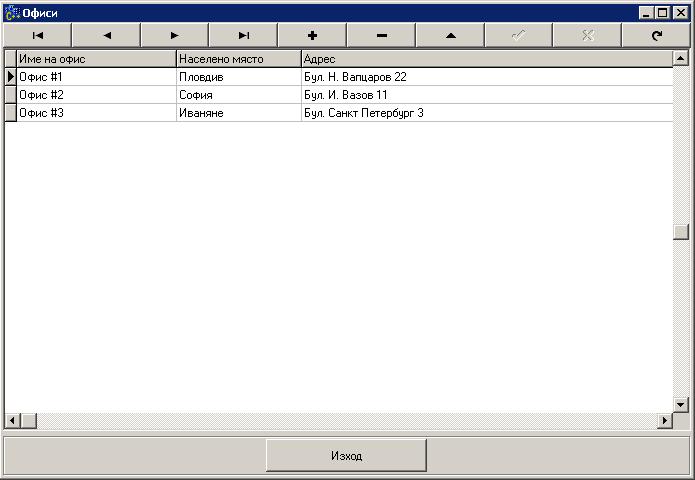
Формата обработва следните събития:

* OrderComboPerson->**OnSelect ()** – настъпва, когато се избере елемент от комбинираната кутия за определяне реда на показване на записи. Променя индекса на таблицата със служители.
* PersonTblSearchEB->**OnChange ()** – настъпва, когато се въведе символ в текстова кутия за търсене. Извършва търсене още в момента на въвеждане.
* PersonTblDBGrid->**OnExit ()** – настъпва, когато grid компонента загуби фокус. Извършва запис на данните в базата данни, ако потребителят не ги е записал, а е напуснал границите на grid-а, докато редактира запис.

**4.6. Описание на форма за редактиране на офиси OfficeFORM**

**4.6.1. Цел и логика на форма за редактиране на офиси OfficeFORM**

Целта на тази форма е да предостави възможност на потребителя или системния администратор да променя, добавя или изтрива записи от таблицата с офиси в базата данни. Формата съдържа бутон за изход, навигатор за записи и grid за визуализиране. Ето как изглежда тя:



**Фигура 16: Форма OfficeFORM**

**4.6.2. Съставни части на форма за редактиране на офиси OfficeFORM**

Формата съдържа следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи от таблицата за служители – OfficeDBGrid
* Бутон за изход от формата – OfficeBitBtn
* Навигатор за записите в таблицата, който позволява добавяне, изтриване и редактиране на записи – OfficeDBNavigator

**4.6.3. Събития на форма за редактиране на офиси OfficeFORM**

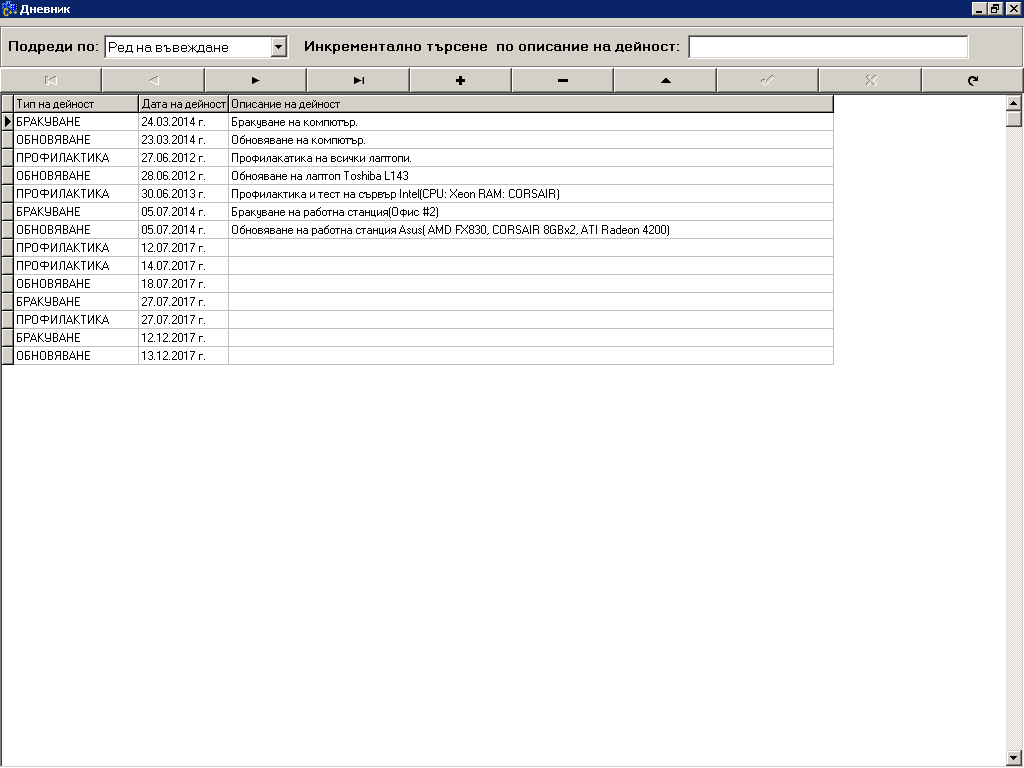
Във формата се обработват следните събития:

* OfficeBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за изход. Затваря формата.
* OfficeDBGrid->OnExit () – настъпва при загуба на фокус от страна на grid-а. Записва данните, които не са записани по време на загуба на фокуса.

**4.7. Описание на форма за редактиране на дейности ActRecFORM**

**4.7.1. Цел и логика на форма за редактиране на дейности ActRecFORM**

Целта на тази форма е да предостави достъп на потребителя или системния администратор до дневника на събития на ИТ фирмата, в който се записва всяко изминало и предстоящо събитие. Формата предоставя способи за редактиране, добавяне и изтриване на събития от дневника. Потребителят може да избира реда на показване на записите или да търси такъв по описание. Ето как изглежда формата:



**Фигура 17: Форма ActRecFORM**

**4.7.2. Съставни части на форма за редактиране на дейности ActRecFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи – ActRecDBGrid
* Компонент от клас TDBNavigator за изтриване, създаване и редактиране на записи – ActRecDBNavigator
* Комбинирана кутия за определяне на реда на показване на записи – ActRecCombo. Тя определя индекса на таблицата с дейности.
* Текстова кутия за търсене на описание –ActivitiesTblSearchEB

**4.7.3. Събития на форма за редактиране на дейност ActRecFORM**

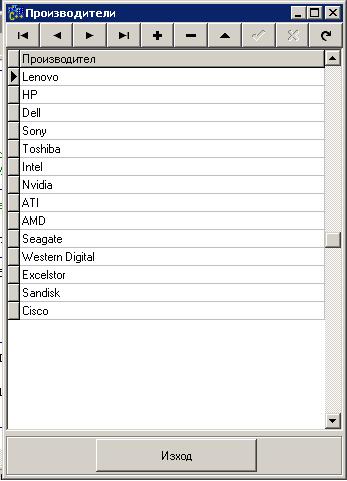
Формата обработва следните събития:

* OrderComboActivities->**OnSelect ()** – настъпва, когато се избере ред от комбинираната кутия. Задава индекса на таблицата с дейностите, който определя и реда на показване на записите.
* ActivitiesTblSearchEB->**OnChange ()** – настъпва, когато се промени съдържанието на текстовата кутия. Извършва инкрементално търсене по описание на дейност.
* ActRecDBGrid->**OnExit ()**–настъпва, когато grid-а загуби фокус. Записва данните, които потребителят редактира, когато grid-а загуби фокус.

**4.8. Описание на форма за редактиране на номенклатура производители ManufactFORM**

**4.8.1. Цел и логика на форма за редактиране ManufactFORM**

Целта на формата е да предостави възможност на потребителя или системния администратор за редактиране на списъка с производители в базата данни. Записите могат да се въвеждат, изтриват или редактират. Ето как изглежда формата:



**Фигура 18: Форма ManufactFORM**

**4.8.2. Съставни части на форма за редактиране ManufactFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи – ManufactDBGrid
* Компонент от клас TDBNavigator за изтриване, създаване и редактиране на записи – ManufactDBNavigator
* Бутон за изход от формата - ManufactBitBtn

**4.8.3. Събития на форма за редактиране ManufactFORM**

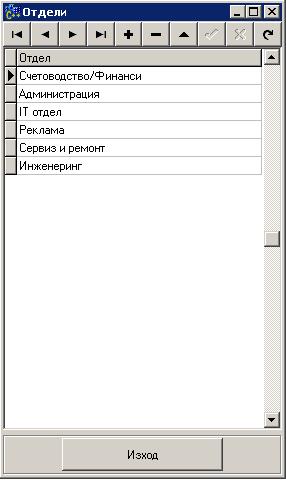
Формата съдържа следните събития:

* ManufactDBGrid->**OnExit ()** – настъпва, когато grid-а загуби фокус. Записва данните на потребителя, ако grid-a загуби фокус, докато е в режим на редактиране.
* ManufacBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за изход. Затваря формата.

**4.9. Описание на форма за редактиране на номенклатура отдели DepFORM**

**4.9.1. Цел и логика на форма за редактиране DepFORM**

Целта на формата е да предостави възможност на потребителя или системния администратор за редактиране на списъка с отдели в базата данни. Записите могат да се въвеждат, изтриват или редактират. Ето как изглежда формата:



**Фигура 19: Форма DepFORM**

**4.9.2. Съставни части на форма за редактиране DepFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи – DepartDBGrid
* Компонент от клас TDBNavigator за изтриване, създаване и редактиране на записи – DepartDBNavigator
* Бутон за изход от формата - DepartBitBtn

**4.9.3. Събития на форма за редактиране DepFORM**

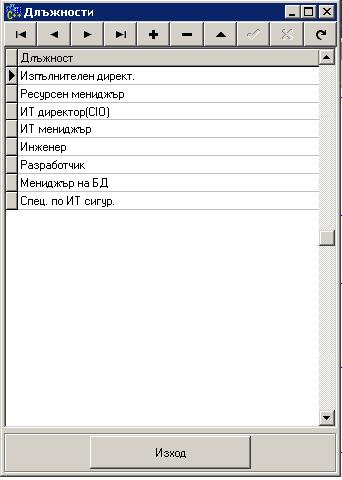
Формата съдържа следните събития:

* DepartDBGrid->**OnExit ()** – настъпва, когато grid-а загуби фокус. Записва данните на потребителя, ако grid-a загуби фокус, докато е в режим на редактиране.
* DepartBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за изход. Затваря формата.

**4.10. Описание на форма за редактиране на номенклатура длъжности PosFORM**

**4.10.1. Цел и логика на форма за редактиране PosFORM**

Целта на формата е да предостави възможност на потребителя или системния администратор за редактиране на списъка с длъжности в базата данни. Записите могат да се въвеждат, изтриват или редактират. Ето как изглежда формата:



**Фигура 20: Форма PosFORM**

**4.10.2. Съставни части на форма за редактиране PosFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи - PositionDBGrid
* Компонент от клас TDBNavigator за изтриване, създаване и редактиране на записи – PositionDBNavigator
* Бутон за изход от формата - PositionBitBtn

**4.10.3. Сибития на форма за редактиране PosFORM**

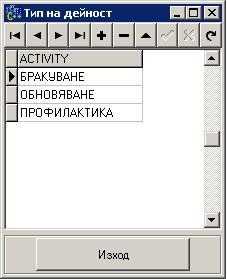
Формата съдържа следните събития:

* PositionDBGrid->**OnExit ()** – настъпва, когато grid-а загуби фокус. Записва данните на потребителя, ако grid-a загуби фокус, докато е в режим на редактиране.
* PositionBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за изход. Затваря формата.

**4.11. Описание на форма за редактиране на номенклатура тип на дейност ActTypeFORM**

**4.11.1. Цел и логика на форма за редактиране ActTypeFORM**

Целта на формата е да предостави възможност на потребителя или системния администратор за редактиране на списъка с типове дейности в базата данни. Записите могат да се въвеждат, изтриват или редактират. Ето как изглежда формата:

****

**Фигура 21: Форма ActTypeFORM**

**4.11.2.Съставни части на форма за редактиране ActTypeFORM**

Формата се състои от следните компоненти:

* Компонент от клас TDBGrid за визуализиране и редактиране на записи - ActTypeDBGrid
* Компонент от клас TDBNavigator за изтриване, създаване и редактиране на записи – ActTypeDBNavigator
* Бутон за изход от формата - ActTypeBitBtn

**4.11.3. Събития на форма за редактиране ActTypeFORM**

Формата съдържа следните събития:

* ActTypeDBGrid->**OnExit ()** – настъпва, когато grid-а загуби фокус. Записва данните на потребителя, ако grid-a загуби фокус, докато е в режим на редактиране.
* ActTypeBitBtn->**OnClick ()** – настъпва, когато потребителят избере бутона за изход. Затваря формата.

**Глава 5**

**Ръководство за потребителя**

**5.1. Добавяне, изтриване и редактиране на компютър**

За да се добави компютър към списъкът с компютри, се следват следните стъпки:

1. Маркира се последният компютър от списъкът с компютри
2. Натиска се клавиатурен клавиш „↓“ („стрелка надолу“) или клавиш Tab (след като сме маркирали последната клетка на последния запис)
3. Попълват се полетата на новият запис и се изпълнява стъпка 2)

За да се изтрие компютър от списъка, трябва да се следват следните стъпки:

1. Маркира се компютърът, който желаем да бъде изтрит
2. От главното меню се избира Редактиране->Компютър->Изтриване

За да се редактира съдържанието на запис с вече съществуващ компютър, трябва да се:

1. Маркира компютърът, който ще редактираме
2. Двоен клик върху клетката, чието съдържание искаме да редактираме
3. Въвеждане на новото съдържание и избиране на клавиш Tab

**5.2. Добавяне, изтриване и редактиране на компонент**

Добавянето, изтриването и редактирането на компонент има два смисъла – (например за добавяне) или да добавим компонент към съществуващ компютър, или да добавим компонент към номенклатурата с компоненти.

За да добавим компонент към номенклатурата с компоненти, трябва:

1. Да се избере от главното меню Редактиране->Компонент->Добавяне
2. Избира се категория на компонента (процесор, RAM памет и т.н.)
3. След това, в самата номенклатура с компоненти от избрания тип натискаме клавиш „↓“ (след като сме маркирали последният запис) или избираме бутон „+“ от навигатора за записи
4. Въвеждаме новото съдържание и натискаме клавиш „↓“ или бутон „✓“от навигатора за записи

За да се изтрие компонент от номенклатурата с компоненти, трябва:

1. Да се избере от главното меню Редактиране->Компонент->Добавяне
2. Избира се категория на компонента (процесор, RAM памет и т.н.)
3. След това, в самата номенклатура с компоненти от избрания тип, избираме компонента, който искаме да премахнем
4. Натискаме бутон „—„ от навигатора за записи

За да се редактира компонент от номенклатурата с компоненти, трябва:

1. Да се избере от главното меню Редактиране->Компонент->Добавяне
2. Избира се категория на компонента (процесор, RAM памет и т.н.)
3. След това, в самата номенклатура с компоненти от избрания тип, избираме компонента, който искаме да редактираме и кликаме два пъти върху клетката, която трябва да се редактира
4. Въвеждаме новото съдържание, натискаме клавиш Enter и след това клавиш „↓“

За да се добави компонент към даден компютър, трябва:

1. Да се маркира компютърът, към който ще се добавя компонент
2. Да се избере от главното меню Редактиране->Компонент->Добавяне
3. Да се избере типът на компонента
4. Да се избере самият компонент
5. Да се зададат номер на фактура и сериен номер на компонента
6. Да се зададат дата на фактура и дата на гаранция
7. Да се натисне бутон ОК от формата

За да се изтрие компонент на даден компютър, трябва:

1. Да се избере компютър
2. Да се избере компонентът, който искаме да изтрием
3. От главното меню да се избере Редактиране->Компонент->Изтрив.

За да се редактира компонент на определен компютър, трябва първо да се изтрие съответният компонент и после да се добави наново с промените данни. Няма директен метод, чрез който може да се редактира компонент на компютър.

**5.3. Добавяне, изтриване и редактиране на принтер**

Стъпките за добавяне, изтриване и редактиране на принтер са аналогични на стъпките за добавяне, изтриване и редактиране на компютър.

**5.4. Добавяне, изтриване и редактиране на комутатор**

Стъпките за добавяне, изтриване и редактиране на комутатор са аналогични на стъпките за добавяне, изтриване и редактиране на компютър.

**5.5.Добавяне, изтриване и редактиране на маршрутизатор**

Стъпките за добавяне, изтриване и редактиране на маршрутизатор са аналогични на стъпките за добавяне, изтриване и редактиране на компютър.

**5.6. Добавяне, изтриване и редактиране на UPS устройство**

Стъпките за добавяне, изтриване и редактиране на UPS устройство са аналогични на стъпките за добавяне, изтриване и редактиране на компютър.

**5.7. Добавяне, изтриване и редактиране на служители**

Стъпките за добавяне, изтриване и редактиране на служител са аналогични на стъпките за добавяне, изтриване и редактиране на компютър с единствената разлика, че от главното меню трябва най-напред да се избере Файл->Служители.

**5.8. Добавяне, изтриване и редактиране на събитие от дневника**

Стъпките за добавяне, изтриване и редактиране на записи от дневника са аналогични на стъпките за добавяне, изтриване и редактиране на компютър с единствената разлика, че от главното меню трябва най-напред да се избере Файл->Дневник.

**5.9. Редактиране на номенклатурите в базата данни**

Стъпките за добавяне, изтриване и редактиране на номенклатури са аналогични на стъпките за добавяне, изтриване и редактиране на компютър с единствената разлика, че от главното меню трябва най-напред да се избере Редактиране->Номенклатури и конкретната номенклатура. Например, за производители пътят до формата изглежда така: Редактиране->Номенклатури->Производители. След това, от показаната форма се излиза с бутона „Изход“ или бутона „х“ в горният десен ъгъл на формата.

**5.10. Съставяне на справки**

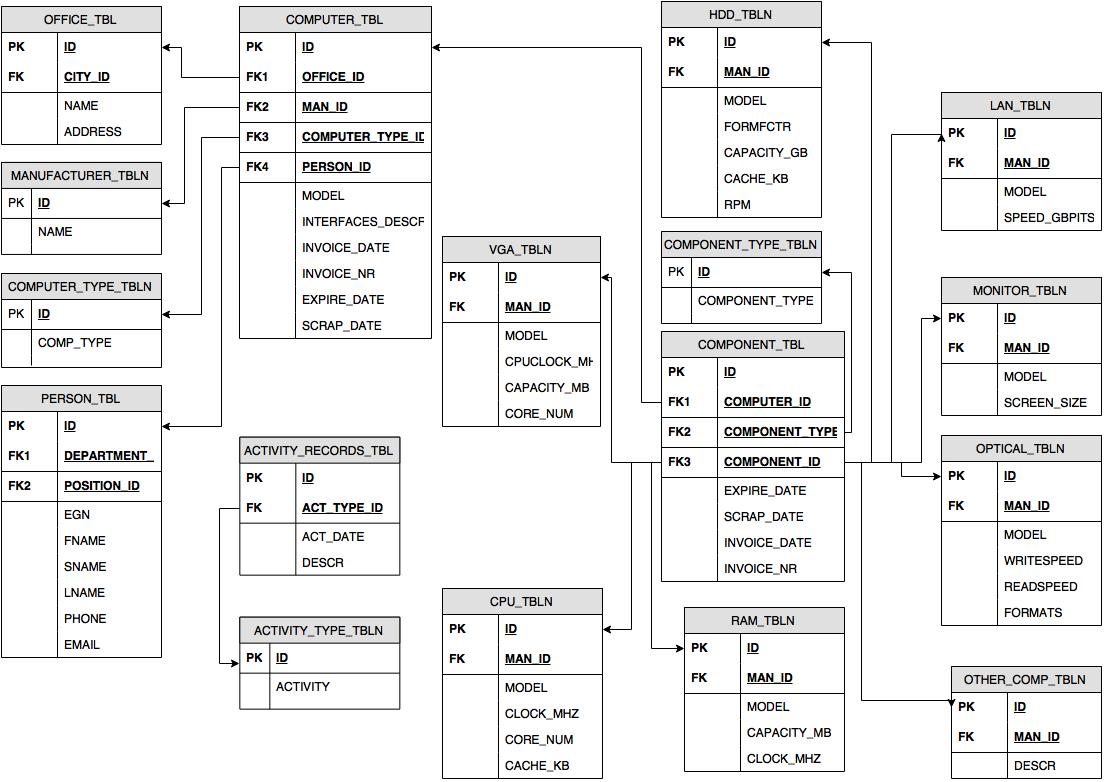
За да се направи справка към базата данни, трябва:

1. Да се избере опция „Справки“ от главното меню
2. Да се изберат параметрите на справката от падащите менюта на комбинираните кутии
3. Да се маркира кои параметри да бъдат взети предвид, като се сложат отметки в съответните кутии
4. Да се въведат/изберат данни от отметнатите кутии
5. Да се натисне бутон „НАПРАВИ СПРАВКА“
6. По желание, може да се извърши експорт на резултатите от направената справка към текстов файл, като се натисне бутон „Експорт към текстов файл“. Преди да бъде избран този бутон, в текстовото поле може да се въведе броят символи, които са позволени на колона т.е. каква е „широчината“ на колоните в текстовия файл.

**Използвана литература**

* Документация на машина за бази данни InterBase - http://docs.embarcadero.com/products/interbase/
* Hollingworth J., Borland C++ Builder 6 Developers Guide, Sams Publishing, 2002, ISBN 0-672-32480-6
* Calvert C. & Phillips J., C++ Builder 3 Unleashed, Sams Publishing, 1998, ISBN0-672-31265-4

**Приложение: Табло със структурата на базата данни**

****

**Приложение: Основни програмни модули и взаимовръзките между тях**

Главна форма на програмата

**Form1**

DataModule

**DModFRM**

Форма за генериране справки

**RefFRM**

Форма за редактиране на дейности

**ActRecFORM**

Форма за редактиране на отдели

**DepFORM**

Форма за редактиране на офиси

**OfficeFORM**

Форма за въвеждане на компоненти

**ComponentEditFORM**

Форма за редактиране на служители

**PersonFORM**

Форма за редактиране на производители

**ManufactFORM**

Форма за редактиране на длъжности

**PosFORM**

Форма за редактиране на тип на дейност

**ActTypeFORM**

**Приложение: SQL скрипт на базата данни**

/\*====================================\*/

**CREATE** DATABASE "C:\Documents and Settings\Administrator\My Documents\Технически университет\Дипломна работа\SysAdmin\_v3.1\Data\IBSysAdmin\_v3.1.gdb"

**user**"SYSDBA"

password "masterkey"

page\_size**=**4096**;**

/\*====================================\*/

**CREATETABLE** MANUFACTURER\_TBLN

**(**

ID **INTEGERNOTNULL,**

NAME **VARCHAR(**50**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** CPU\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

CLOCK\_MHZ **INTEGER,**

CORE\_NUM **INTEGER,**

CACHE\_KB **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** HDD\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

FORMFCTR **FLOAT,**

CAPACITY\_GB **INTEGER,**

CACHE\_KB **INTEGER,**

RPM **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** RAM\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

CAPACITY\_MB **INTEGER,**

CLOCK\_MHZ **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** LAN\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

SPEED\_GBPITS **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** VGA\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

CAPACITY\_MB **INTEGER,**

CPUCLOCK\_MHZ **INTEGER,**

CORE\_NUM **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** OPTICAL\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

WRITESPEED **INTEGER,**

READSPEED **INTEGER,**

FORMATS **VARCHAR(**50**),**/\*DVD,CD,VCD...\*/

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** MONITOR\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

SCREEN\_SIZE **FLOAT,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** OTHER\_COMP\_TBLN

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

DESCR **VARCHAR(**200**),**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** COMPUTER\_TYPE\_TBLN

**(**

ID **INTEGERNOTNULL,**

COMP\_TYPE **VARCHAR(**10**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** COMPONENT\_TYPE\_TBLN

**(**

ID **INTEGERNOTNULL,**

COMPONENT\_TYPE **VARCHAR(**20**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** POSITION\_TBLN

**(**

ID **INTEGERNOTNULL,**

NAME **VARCHAR(**20**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** DEPARTMENT\_TBLN

**(**

ID **INTEGERNOTNULL,**

NAME **VARCHAR(**20**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** ACTIVITY\_TYPE\_TBLN

**(**

ID **INTEGERNOTNULL,**

ACTIVITY **VARCHAR(**20**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** ACTIVITY\_RECORDS\_TBL

**(**

ID **INTEGERNOTNULL,**

ACT\_TYPE\_ID **INTEGER,**

ACT\_DATE **DATE,**

DESCR **VARCHAR(**100**),**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**ACT\_TYPE\_ID**)REFERENCES** ACTIVITY\_TYPE\_TBLN**(**ID**)**

**);**

**CREATETABLE** REGION\_TBLN

**(**

ID **INTEGERNOTNULL,**

NAME **VARCHAR(**20**),**

**PRIMARYKEY(**ID**)**

**);**

**CREATETABLE** MUNICIPALITY\_TBLN

**(**

ID **INTEGERNOTNULL,**

NAME **VARCHAR(**20**),**

REGION\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**REGION\_ID**)REFERENCES** REGION\_TBLN**(**ID**)**

**);**

**CREATETABLE** CITY\_TBLN

**(**

ID **INTEGERNOTNULL,**

EKATTE **INTEGERNOTNULLUNIQUE,**

NAME **VARCHAR(**20**),**

MUNICIPALITY\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**MUNICIPALITY\_ID**)REFERENCES** MUNICIPALITY\_TBLN**(**ID**)**

**);**

**CREATETABLE** OFFICE\_TBL

**(**

ID **INTEGERNOTNULL,**

CITY\_ID **INTEGER,**

NAME **VARCHAR(**50**),**

ADDRESS **VARCHAR(**100**),**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**CITY\_ID**)REFERENCES** CITY\_TBLN**(**ID**)**

**);**

**CREATETABLE** PERSON\_TBL

**(**

ID **INTEGERNOTNULL,**

EGN **VARCHAR(**20**),**

FNAME **VARCHAR(**20**),**

SNAME **VARCHAR(**20**),**

LNAME **VARCHAR(**20**),**

PHONE **VARCHAR(**20**),**

EMAIL **VARCHAR(**50**),**

POSITION\_ID **INTEGER,**

DEPARTMENT\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**POSITION\_ID**)REFERENCES** POSITION\_TBLN**(**ID**),**

**FOREIGNKEY(**DEPARTMENT\_ID**)REFERENCES** DEPARTMENT\_TBLN**(**ID**)**

**);**

**CREATETABLE** UPS\_TBL

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

MAX\_STATIONS **INTEGER,**

MAX\_MINUTES **INTEGER,**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

EXPIRE\_DATE **DATE,**

SCRAP\_DATE **DATE,**

OFFICE\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**OFFICE\_ID**)REFERENCES** OFFICE\_TBL**(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** SWITCH\_TBL

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

SPEED\_GBIT **INTEGER,**

PORT\_NUM **INTEGER,**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

EXPIRE\_DATE **DATE,**

SCRAP\_DATE **DATE,**

OFFICE\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**OFFICE\_ID**)REFERENCES** OFFICE\_TBL**(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** PRINTER\_TBL

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

PAGES\_PER\_MIN **INTEGER,**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

EXPIRE\_DATE **DATE,**

SCRAP\_DATE **DATE,**

OFFICE\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**OFFICE\_ID**)REFERENCES** OFFICE\_TBL**(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** ROUTER\_TBL

**(**

ID **INTEGERNOTNULL,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

MAX\_STATIONS **INTEGER,**

RANGE\_METERS **INTEGER,**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

EXPIRE\_DATE **DATE,**

SCRAP\_DATE **DATE,**

OFFICE\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**OFFICE\_ID**)REFERENCES** OFFICE\_TBL**(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**)**

**);**

**CREATETABLE** COMPUTER\_TBL

**(**

ID **INTEGERNOTNULL,**

COMPUTER\_TYPE\_ID **INTEGER,**

MAN\_ID **INTEGER,**

MODEL **VARCHAR(**50**),**

INTERFACES\_DESCR **VARCHAR(**100**),**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

EXPIRE\_DATE **DATE,**

SCRAP\_DATE **DATE,**

OFFICE\_ID **INTEGER,**

PERSON\_ID **INTEGER,**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**COMPUTER\_TYPE\_ID**)REFERENCES** COMPUTER\_TYPE\_TBLN**(**ID**),**

**FOREIGNKEY(**OFFICE\_ID**)REFERENCES** OFFICE\_TBL**(**ID**),**

**FOREIGNKEY(**MAN\_ID**)REFERENCES** MANUFACTURER\_TBLN**(**ID**),**

**FOREIGNKEY(**PERSON\_ID**)REFERENCES** PERSON\_TBL**(**ID**)**

**);**

**CREATETABLE** COMPONENT\_TBL

**(**

ID **INTEGERNOTNULL,**

COMPUTER\_ID **INTEGER,**

COMPONENT\_TYPE\_ID **INTEGER,**

COMPONENT\_ID **INTEGER,**

EXPIRE\_DATE **DATE,**

INVOICE\_DATE **DATE,**

INVOICE\_NR **INTEGER,**

SERIAL\_NR **VARCHAR(**40**),**

**PRIMARYKEY(**ID**),**

**FOREIGNKEY(**COMPUTER\_ID**)REFERENCES** COMPUTER\_TBL**(**ID**),**

**FOREIGNKEY(**COMPONENT\_TYPE\_ID**)REFERENCES** COMPONENT\_TYPE\_TBLN**(**ID**)**

**);**

/\*GENERATORS\*/

**CREATE** GENERATOR ACTRECTBL\_IDGEN**;**

**SET** GENERATOR ACTRECTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR COMPONENTTBL\_IDGEN**;**

**SET** GENERATOR COMPONENTTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR COMPUTERTBL\_IDGEN**;**

**SET** GENERATOR COMPUTERTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR OFFICETBL\_IDGEN**;**

**SET** GENERATOR OFFICETBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR PERSONTBL\_IDGEN**;**

**SET** GENERATOR PERSONTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR PRINTERTBL\_IDGEN**;**

**SET** GENERATOR PRINTERTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR ROUTERTBL\_IDGEN**;**

**SET** GENERATOR ROUTERTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR SWITCHTBL\_IDGEN**;**

**SET** GENERATOR SWITCHTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR UPSTBL\_IDGEN**;**

**SET** GENERATOR UPSTBL\_IDGEN **TO**0**;**

**CREATE** GENERATOR ACTTYPETBLN\_IDGEN**;**

**SET** GENERATOR ACTTYPETBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR CITYTBLN\_IDGEN**;**

**SET** GENERATOR CITYTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR COMPONENTTYPETBLN\_IDGEN**;**

**SET** GENERATOR COMPONENTTYPETBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR COMPUTERTYPETBLN\_IDGEN**;**

**SET** GENERATOR COMPUTERTYPETBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR CPUTBLN\_IDGEN**;**

**SET** GENERATOR CPUTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR DEPTBLN\_IDGEN**;**

**SET** GENERATOR DEPTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR HDDTBLN\_IDGEN**;**

**SET** GENERATOR HDDTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR LANTBLN\_IDGEN**;**

**SET** GENERATOR LANTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR MANTBLN\_IDGEN**;**

**SET** GENERATOR MANTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR MONTBLN\_IDGEN**;**

**SET** GENERATOR MONTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR MUNITBLN\_IDGEN**;**

**SET** GENERATOR MUNITBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR OPTTBLN\_IDGEN**;**

**SET** GENERATOR OPTTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR OTHERTBLN\_IDGEN**;**

**SET** GENERATOR OTHERTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR POSTBLN\_IDGEN**;**

**SET** GENERATOR POSTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR RAMTBLN\_IDGEN**;**

**SET** GENERATOR RAMTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR REGIONTBLN\_IDGEN**;**

**SET** GENERATOR REGIONTBLN\_IDGEN **TO**0**;**

**CREATE** GENERATOR VGATBLN\_IDGEN**;**

**SET** GENERATOR VGATBLN\_IDGEN **TO**0**;**

/\*STORED PROCEDURES\*/

**SET** TERM **^;**

**CREATEPROCEDURE** ACTRECTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**ACTRECTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** COMPONENTTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**COMPONENTTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** COMPUTERTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**COMPUTERTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** OFFICETBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**OFFICETBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** PERSONTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**PERSONTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** PRINTERTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**PRINTERTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** ROUTERTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**ROUTERTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** SWITCHTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**SWITCHTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** UPSTBL\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**UPSTBL\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** ACTTYPETBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**ACTTYPETBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** CITYTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**CITYTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** COMPONENTTYPETBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**COMPONENTTYPETBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** COMPUTERTYPETBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**COMPUTERTYPETBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** CPUTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**CPUTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** DEPTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**DEPTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** HDDTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**HDDTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** LANTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**LANTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** MANTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**MANTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** MONTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**MONTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** MUNITBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**MUNITBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** OPTTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**OPTTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** OTHERTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**OTHERTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** POSTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**POSTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** RAMTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**RAMTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** REGIONTBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**REGIONTBLN\_IDGEN**,**1**);**

**END^**

**CREATEPROCEDURE** VGATBLN\_IDSP

**RETURNS(**ID **INTEGER)**

**AS**

**BEGIN**

ID **=** GEN\_ID**(**VGATBLN\_IDGEN**,**1**);**

**END^**

/\*TABLE INDEXES\*/

**CREATEINDEX** ActRecTblNDX **ON** ACTIVITY\_RECORDS\_TBL**(**ACT\_DATE**);**

**CREATEINDEX** CityTblnNDX **ON** CITY\_TBLN**(**NAME**);**

**CREATEINDEX** ManufactTblNameNDX **ON** MANUFACTURER\_TBLN**(**NAME**);**

**CREATEINDEX** OfficeTblNameNDX **ON** OFFICE\_TBL**(**NAME**);**

**CREATEINDEX** CmpntTblExpDateNDX **ON** COMPONENT\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** CmpntTblInvDateNDX **ON** COMPONENT\_TBL**(**INVOICE\_DATE**);**

**CREATEINDEX** CmptrTblScrapNDX **ON** COMPUTER\_TBL**(**SCRAP\_DATE**);**

**CREATEINDEX** CmptrTblExpDateNDX **ON** COMPUTER\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** CmptrTblModelNDX **ON** COMPUTER\_TBL**(**MODEL**);**

**CREATEINDEX** CmptrTblInvDateNDX **ON** COMPUTER\_TBL**(**INVOICE\_DATE**);**

**CREATEINDEX** CmptrTblInvNrNDX **ON** COMPUTER\_TBL**(**INVOICE\_NR**);**

**CREATEINDEX** MunicipTblNameNDX **ON** MUNICIPALITY\_TBLN**(**NAME**);**

**CREATEINDEX** PersonTblFNameNDX **ON** PERSON\_TBL**(**FNAME**);**

**CREATEINDEX** PersonTblLNameNDX **ON** PERSON\_TBL**(**LNAME**);**

**CREATEINDEX** PersonTblSNameNDX **ON** PERSON\_TBL**(**SNAME**);**

**CREATEINDEX** PersonTblEgnNDX **ON** PERSON\_TBL**(**EGN**);**

**CREATEINDEX** PersonTblPhoneNDX **ON** PERSON\_TBL**(**PHONE**);**

**CREATEINDEX** RegionTblnNameNDX **ON** REGION\_TBLN**(**NAME**);**

**CREATEINDEX** RouterTblExpDateNDX **ON** ROUTER\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** RouterTblScrapNDX **ON** ROUTER\_TBL**(**SCRAP\_DATE**);**

**CREATEINDEX** RouterTblModelNDX **ON** ROUTER\_TBL**(**MODEL**);**

**CREATEINDEX** RouterTblInvDateNDX **ON** ROUTER\_TBL**(**INVOICE\_DATE**);**

**CREATEINDEX** SwitchTblExpDateNDX **ON** SWITCH\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** SwitchTblScrapNDX **ON** SWITCH\_TBL**(**SCRAP\_DATE**);**

**CREATEINDEX** SwitchTblModelNDX **ON** SWITCH\_TBL**(**MODEL**);**

**CREATEINDEX** SwitchTblInvDateNDX **ON** SWITCH\_TBL**(**INVOICE\_DATE**);**

**CREATEINDEX** UpsTblExpDateNDX **ON** UPS\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** UpsTblScrapNDX **ON** UPS\_TBL**(**SCRAP\_DATE**);**

**CREATEINDEX** UpsTblModelNDX **ON** UPS\_TBL**(**MODEL**);**

**CREATEINDEX** UpsTblInvDateNDX **ON** UPS\_TBL**(**INVOICE\_DATE**);**

**CREATEINDEX** PrinterTblScrapNDX **ON** PRINTER\_TBL**(**SCRAP\_DATE**);**

**CREATEINDEX** PrinterTblExpDateNDX **ON** PRINTER\_TBL**(**EXPIRE\_DATE**);**

**CREATEINDEX** PrinterTblModelNDX **ON** PRINTER\_TBL**(**MODEL**);**

**CREATEINDEX** PrinterTblInvDateNDX **ON** PRINTER\_TBL**(**INVOICE\_DATE**);**

/\*ADD COLUM CONSTRAINTS ON DELETE AND ON UPDATE\*/

**ALTERTABLE** COMPONENT\_TBL **ADDFOREIGNKEY(**COMPUTER\_ID**)**

**REFERENCES** COMPUTER\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** COMPONENT\_TBL **ADDFOREIGNKEY(**COMPONENT\_TYPE\_ID**)**

**REFERENCES** COMPONENT\_TYPE\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** ACTIVITY\_RECORDS\_TBL **ADDFOREIGNKEY(**ACT\_TYPE\_ID**)**

**REFERENCES** ACTIVITY\_TYPE\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** CITY\_TBLN **ADDFOREIGNKEY(**MUNICIPALITY\_ID**)**

**REFERENCES** MUNICIPALITY\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** MUNICIPALITY\_TBLN **ADDFOREIGNKEY(**REGION\_ID**)**

**REFERENCES** REGION\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** COMPUTER\_TBL **ADDFOREIGNKEY(**COMPUTER\_TYPE\_ID**)**

**REFERENCES** COMPUTER\_TYPE\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** COMPUTER\_TBL **ADDFOREIGNKEY(**OFFICE\_ID**)**

**REFERENCES** OFFICE\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** COMPUTER\_TBL **ADDFOREIGNKEY(**PERSON\_ID**)**

**REFERENCES** PERSON\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** CPU\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** VGA\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** LAN\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** HDD\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** MONITOR\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** OTHER\_COMP\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** RAM\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** OPTICAL\_TBLN **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** OFFICE\_TBL **ADDFOREIGNKEY(**CITY\_ID**)**

**REFERENCES** CITY\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** PERSON\_TBL **ADDFOREIGNKEY(**POSITION\_ID**)**

**REFERENCES** POSITION\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** PERSON\_TBL **ADDFOREIGNKEY(**DEPARTMENT\_ID**)**

**REFERENCES** DEPARTMENT\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** PRINTER\_TBL **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** PRINTER\_TBL **ADDFOREIGNKEY(**OFFICE\_ID**)**

**REFERENCES** OFFICE\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** SWITCH\_TBL **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** SWITCH\_TBL **ADDFOREIGNKEY(**OFFICE\_ID**)**

**REFERENCES** OFFICE\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** UPS\_TBL **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** UPS\_TBL **ADDFOREIGNKEY(**OFFICE\_ID**)**

**REFERENCES** OFFICE\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** ROUTER\_TBL **ADDFOREIGNKEY(**MAN\_ID**)**

**REFERENCES** MANUFACTURER\_TBLN**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

**ALTERTABLE** ROUTER\_TBL **ADDFOREIGNKEY(**OFFICE\_ID**)**

**REFERENCES** OFFICE\_TBL**(**ID**)ONDELETECASCADEONUPDATECASCADE;**

/\*CHANGE THE COMPONENT INFO STORED PROCEDURES\*/

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_CPU\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** mname **VARCHAR(**50**);**

**DECLAREVARIABLE** cmodel **VARCHAR(**50**);**

**DECLAREVARIABLE** cclock **INTEGER;**

**DECLAREVARIABLE** ccore **INTEGER;**

**DECLAREVARIABLE** ccache **INTEGER;**

**BEGIN**

**FORSELECT** M**.**NAME**,** C**.**MODEL**,** C**.**CLOCK\_MHZ**,** C**.**CORE\_NUM**,** C**.**CACHE\_KB

**FROM** CPU\_TBLN C **JOIN** MANUFACTURER\_TBLN M **ON** C**.**MAN\_ID **=** M**.**ID

**WHERE** C**.**ID **=:**CT\_ID

**INTO:**mname**,:**cmodel**,:**cclock**,:**ccore**,:**ccache

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**mname **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**mname **||**'; '**;**

**IF(:**cmodel **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**cmodel **||**'; '**;**

**IF(:**cclock **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**cclock **ASVARCHAR(**20**)**)**||**' MHz; '**;**

**IF(:**ccore **ISNOTNULL**)**THEN**

**IF(:**ccore **>**1)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**ccore **ASVARCHAR(**20**)**)**||**' cores; '**;**

**ELSE**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**ccore **ASVARCHAR(**20**)**)**||**' core; '**;**

**IF(:**ccache **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**ccache **ASVARCHAR(**20**)**)**||**' KB; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_VGA\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** mname **VARCHAR(**50**);**

**DECLAREVARIABLE** vmodel **VARCHAR(**50**);**

**DECLAREVARIABLE** vclock **INTEGER;**

**DECLAREVARIABLE** vcore **INTEGER;**

**DECLAREVARIABLE** vcapacity **INTEGER;**

**BEGIN**

**FORSELECT** M**.**NAME**,** V**.**MODEL**,** V**.**CPUCLOCK\_MHZ**,** V**.**CORE\_NUM**,** V**.**CAPACITY\_MB

**FROM** VGA\_TBLN V **JOIN** MANUFACTURER\_TBLN M **ON** V**.**MAN\_ID **=** M**.**ID

**WHERE** V**.**ID **=:**CT\_ID

**INTO:**mname**,:**vmodel**,:**vclock**,:**vcore**,:**vcapacity

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**mname **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**mname **||**'; '**;**

**IF(:**vmodel **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**vmodel **||**'; '**;**

**IF(:**vclock **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**vclock **ASVARCHAR(**20**)**)**||**' MHz; '**;**

**IF(:**vcore **ISNOTNULL**)**THEN**

**IF(:**vcore **>**1)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**vcore **ASVARCHAR(**20**)**)**||**' cores; '**;**

**ELSE**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**vcore **ASVARCHAR(**20**)**)**||**' core; '**;**

**IF(:**vcapacity **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**vcapacity **ASVARCHAR(**20**)**)**||**' MB; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_HDD\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** mname **VARCHAR(**50**);**

**DECLAREVARIABLE** hmodel **VARCHAR(**50**);**

**DECLAREVARIABLE** hform **FLOAT;**

**DECLAREVARIABLE** hcap **INTEGER;**

**DECLAREVARIABLE** hcache **INTEGER;**

**DECLAREVARIABLE** hrpm **INTEGER;**

**BEGIN**

**FORSELECT** M**.**NAME**,** H**.**MODEL**,** H**.**FORMFCTR**,** H**.**CAPACITY\_GB**,** H**.**CACHE\_KB**,** H**.**RPM

**FROM** HDD\_TBLN H **JOIN** MANUFACTURER\_TBLN M **ON** H**.**MAN\_ID **=** M**.**ID

**WHERE** H**.**ID **=:**CT\_ID

**INTO:**mname**,:**hmodel**,:**hform**,:**hcap**,:**hcache**,:**hrpm

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**mname **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**mname **||**'; '**;**

**IF(:**hmodel **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**hmodel **||**'; '**;**

**IF(:**hform **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**hform **ASVARCHAR(**3**)**)**||**'"; '**;**

**IF(:**hcap **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**hcap **ASVARCHAR(**20**)**)**||**' GB; '**;**

**IF(:**hcache **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**hcache **ASVARCHAR(**20**)**)**||**' KB; '**;**

**IF(:**hrpm **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**hrpm **ASVARCHAR(**20**)**)**||**' RPM; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_MONITOR\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** mname **VARCHAR(**50**);**

**DECLAREVARIABLE** mmodel **VARCHAR(**50**);**

**DECLAREVARIABLE** msize **FLOAT;**

**BEGIN**

**FORSELECT** M**.**NAME**,** MON**.**MODEL**,** MON**.**SCREEN\_SIZE

**FROM** MONITOR\_TBLN MON **JOIN** MANUFACTURER\_TBLN M **ON** MON**.**MAN\_ID **=** M**.**ID

**WHERE** MON**.**ID **=:**CT\_ID

**INTO:**mname**,:**mmodel**,:**msize

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**mname **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**mname **||**'; '**;**

**IF(:**mmodel **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**mmodel **||**'; '**;**

**IF(:**msize **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**msize **ASVARCHAR(**4**)**)**||**'"; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_LAN\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** name **VARCHAR(**50**);**

**DECLAREVARIABLE** model **VARCHAR(**50**);**

**DECLAREVARIABLE** speed **INTEGER;**

**BEGIN**

**FORSELECT** M**.**NAME**,** L**.**MODEL**,** L**.**SPEED\_GBPITS

**FROM** LAN\_TBLN L **JOIN** MANUFACTURER\_TBLN M **ON** L**.**MAN\_ID **=** M**.**ID

**WHERE** L**.**ID **=:**CT\_ID

**INTO:**name**,:**model**,:**speed

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**name **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**name **||**'; '**;**

**IF(:**model **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**model **||**'; '**;**

**IF(:**speed **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**speed **ASVARCHAR(**20**)**)**||**' Gbits/s; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_OPTICAL\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** name **VARCHAR(**50**);**

**DECLAREVARIABLE** model **VARCHAR(**50**);**

**DECLAREVARIABLE** writespeed **INTEGER;**

**DECLAREVARIABLE** readspeed **INTEGER;**

**DECLAREVARIABLE** formats **VARCHAR(**50**);**

**BEGIN**

**FORSELECT** M**.**NAME**,** O**.**MODEL**,** O**.**WRITESPEED**,** O**.**READSPEED**,** O**.**FORMATS

**FROM** OPTICAL\_TBLN O **JOIN** MANUFACTURER\_TBLN M **ON** O**.**MAN\_ID **=** M**.**ID

**WHERE** O**.**ID **=:**CT\_ID

**INTO:**name**,:**model**,:**writespeed**,:**readspeed**,:**formats

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**name **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**name **||**'; '**;**

**IF(:**model **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**model **||**'; '**;**

**IF(:**writespeed **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**writespeed **ASVARCHAR(**20**)**)**||**'x; '**;**

**IF(:**readspeed **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||CAST(:**readspeed **ASVARCHAR(**20**)**)**||**'x; '**;**

**IF(:**formats **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||**'Formats: '**||:**formats **||**'; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_OTHER\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** name **VARCHAR(**50**);**

**DECLAREVARIABLE** descr **VARCHAR(**200**);**

**BEGIN**

**FORSELECT** M**.**NAME**,** O**.**DESCR

**FROM** OTHER\_COMP\_TBLN O **JOIN** MANUFACTURER\_TBLN M **ON** O**.**MAN\_ID **=** M**.**ID

**WHERE** O**.**ID **=:**CT\_ID

**INTO:**name**,:**descr

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**name **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**name **||**'; '**;**

**IF(:**descr **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**descr **||**'; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_RAM\_INFO\_SP**(**CT\_ID **INTEGER**)

**RETURNS(**COMPONENT\_DESCR **VARCHAR(**512**)**)

**AS**

**DECLAREVARIABLE** name **VARCHAR(**50**);**

**DECLAREVARIABLE** model **VARCHAR(**50**);**

**DECLAREVARIABLE** capacity **INTEGER;**

**DECLAREVARIABLE** clock **INTEGER;**

**BEGIN**

**FORSELECT** M**.**NAME**,** R**.**MODEL**,** R**.**CAPACITY\_MB**,** R**.**CLOCK\_MHZ

**FROM** RAM\_TBLN R **JOIN** MANUFACTURER\_TBLN M **ON** R**.**MAN\_ID **=** M**.**ID

**WHERE** R**.**ID **=:**CT\_ID

**INTO:**name**,:**model**,:**capacity**,:**clock

**DO**

**BEGIN**

COMPONENT\_DESCR **=**''**;**

**IF(:**name **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=:**name **||**'; '**;**

**IF(:**model **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**model **||**'; '**;**

**IF(:**capacity **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**capacity **||**' MB; '**;**

**IF(:**clock **ISNOTNULL**)**THEN**

COMPONENT\_DESCR **=** COMPONENT\_DESCR **||:**clock **||**' MHz; '**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**SET** TERM **^;**

**CREATEPROCEDURE** GET\_PC\_INFO\_SP**(**COMPUTER\_ID **INTEGER**)

**RETURNS(**ID **INTEGER,**

COMPONENT\_ID **INTEGER,**

COMPONENT\_TYPE\_ID **INTEGER,**

COMPONENT\_TYPE **VARCHAR(**20**),**

COMPONENT\_DESCR **VARCHAR(**512**),**

SERIAL\_NR **VARCHAR(**40**),**

INVOICE\_NR **INTEGER,**

INVOICE\_DATE **DATE,**

EXPIRE\_DATE **DATE)**

**AS**

**BEGIN**

**FORSELECT**

C**.**ID**,**

C**.**COMPONENT\_ID**,**

C**.**COMPONENT\_TYPE\_ID**,**

T**.**COMPONENT\_TYPE**,**

C**.**SERIAL\_NR**,**

C**.**INVOICE\_NR**,**

C**.**INVOICE\_DATE**,**

C**.**EXPIRE\_DATE

**FROM** COMPONENT\_TBL C **JOIN** COMPONENT\_TYPE\_TBLN T **ON** C**.**COMPONENT\_TYPE\_ID **=** T**.**ID

**WHERE** C**.**COMPUTER\_ID **=:**COMPUTER\_ID

**INTO:**ID**,**

**:**COMPONENT\_ID**,**

**:**COMPONENT\_TYPE\_ID**,**

**:**COMPONENT\_TYPE**,**

**:**SERIAL\_NR**,**

**:**INVOICE\_NR**,**

**:**INVOICE\_DATE**,**

**:**EXPIRE\_DATE

**DO**

**BEGIN**

**IF(:**COMPONENT\_TYPE\_ID **=**1)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_VGA\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**2**)THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_LAN\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**3)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_CPU\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**4)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_HDD\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**5)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_MONITOR\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**6)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_OPTICAL\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**7)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_OTHER\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

**ELSEIF(:**COMPONENT\_TYPE\_ID **=**8)**THEN**

**SELECT** COMPONENT\_DESCR **FROM** GET\_RAM\_INFO\_SP**(:**COMPONENT\_ID)**INTO:**COMPONENT\_DESCR**;**

SUSPEND**;**

**END**

**END^**

**SET** TERM **;^**

**Приложение: Сорс код на програмното осигуряване**

**DModFRM.h**

#ifndef DModFRMH

#define DModFRMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DB.hpp>

#include <IBDatabase.hpp>

#include <IBCustomDataSet.hpp>

#include <IBTable.hpp>

#include <IBStoredProc.hpp>

#include <IBQuery.hpp>

//---------------------------------------------------------------------------

class TIBDataModule **:**public TDataModule

**{**

\_\_published**:**// IDE-managed Components

TIBDatabase **\***IBDBSysAdmin**;**

TIBTransaction **\***IBTSSysAdmin**;**

TIBTable **\***IBActRecTbl**;**

TIBTable **\***IBActTypeTbln**;**

TIBTable **\***IBCityTbln**;**

TIBTable **\***IBCmpntTbl**;**

TIBTable **\***IBCmpntTypeTbln**;**

TIBTable **\***IBCmptrTbl**;**

TIBTable **\***IBCmptrTypeTbln**;**

TIBTable **\***IBCpuTbln**;**

TIBTable **\***IBDepartTbln**;**

TIBTable **\***IBHddTbln**;**

TIBTable **\***IBLanTbln**;**

TIBTable **\***IBManufactTbln**;**

TIBTable **\***IBMonitorTbln**;**

TIBTable **\***IBMunicipTbln**;**

TIBTable **\***IBOfficeTbl**;**

TIBTable **\***IBOpticalTbln**;**

TIBTable **\***IBOtherTbln**;**

TIBTable **\***IBPersonTbl**;**

TIBTable **\***IBPositionTbln**;**

TIBTable **\***IBPrinterTbl**;**

TIBTable **\***IBRamTbln**;**

TIBTable **\***IBRegionTbln**;**

TIBTable **\***IBRouterTbl**;**

TIBTable **\***IBSwitchTbl**;**

TIBTable **\***IBUpsTbl**;**

TIBTable **\***IBVgaTbln**;**

TIBStoredProc **\***IBActRecIDSP**;**

TIBStoredProc **\***IBActTypeIDSP**;**

TIBStoredProc **\***IBCityIDSP**;**

TIBStoredProc **\***IBCmpntIDSP**;**

TIBStoredProc **\***IBCmpntTypeIDSP**;**

TIBStoredProc **\***IBCmptrIDSP**;**

TIBStoredProc **\***IBCmptrTypeIDSP**;**

TIBStoredProc **\***IBCpuIDSP**;**

TIBStoredProc **\***IBDepartIDSP**;**

TIBStoredProc **\***IBHddIDSP**;**

TIBStoredProc **\***IBLanIDSP**;**

TIBStoredProc **\***IBRouterIDSP**;**

TIBStoredProc **\***IBSwitchIDSP**;**

TIBStoredProc **\***IBManufactIDSP**;**

TIBStoredProc **\***IBMonitorIDSP**;**

TIBStoredProc **\***IBMunicipIDSP**;**

TIBStoredProc **\***IBOfficeIDSP**;**

TIBStoredProc **\***IBOpticalIDSP**;**

TIBStoredProc **\***IBOtherIDSP**;**

TIBStoredProc **\***IBPersonIDSP**;**

TIBStoredProc **\***IBPositionIDSP**;**

TIBStoredProc **\***IBPrinterIDSP**;**

TIBStoredProc **\***IBRamIDSP**;**

TIBStoredProc **\***IBRegionIDSP**;**

TIBStoredProc **\***IBVgaIDSP**;**

TIBStoredProc **\***IBUpsIDSP**;**

TDataSource **\***IBActRecDS**;**

TDataSource **\***IBActTypeDS**;**

TDataSource **\***IBCityDS**;**

TDataSource **\***IBCmpntDS**;**

TDataSource **\***IBCmpntTypeDS**;**

TDataSource **\***IBCmptrDS**;**

TDataSource **\***IBCmptrTypeDS**;**

TDataSource **\***IBCpuDS**;**

TDataSource **\***IBDepartDS**;**

TDataSource **\***IBHddDS**;**

TDataSource **\***IBLanDS**;**

TDataSource **\***IBSwitchDS**;**

TDataSource **\***IBRouterDS**;**

TDataSource **\***IBManufactDS**;**

TDataSource **\***IBMonitorDS**;**

TDataSource **\***IBMunicipDS**;**

TDataSource **\***IBOfficeDS**;**

TDataSource **\***IBOpticalDS**;**

TDataSource **\***IBOtherDS**;**

TDataSource **\***IBPersonDS**;**

TDataSource **\***IBPositionDS**;**

TDataSource **\***IBPrinterDS**;**

TDataSource **\***IBRamDS**;**

TDataSource **\***IBUpsDS**;**

TDataSource **\***IBRegionDS**;**

TDataSource **\***IBVgaDS**;**

TStringField **\***IBCmptrTblCmptrTypeLU**;**

TIntegerField **\***IBCmptrTblID**;**

TIntegerField **\***IBCmptrTblCOMPUTER\_TYPE\_ID**;**

TIntegerField **\***IBCmptrTblMAN\_ID**;**

TIBStringField **\***IBCmptrTblMODEL**;**

TIBStringField **\***IBCmptrTblINTERFACES\_DESCR**;**

TDateField **\***IBCmptrTblINVOICE\_DATE**;**

TIntegerField **\***IBCmptrTblINVOICE\_NR**;**

TDateField **\***IBCmptrTblEXPIRE\_DATE**;**

TDateField **\***IBCmptrTblSCRAP\_DATE**;**

TIntegerField **\***IBCmptrTblOFFICE\_ID**;**

TIntegerField **\***IBCmptrTblPERSON\_ID**;**

TStringField **\***IBCmptrTblManufactLU**;**

TStringField **\***IBCmptrTblPersonLU**;**

TStringField **\***IBCmptrTblOfficeLU**;**

TIntegerField **\***IBActRecTblID**;**

TIntegerField **\***IBActRecTblACT\_TYPE\_ID**;**

TDateField **\***IBActRecTblACT\_DATE**;**

TIBStringField **\***IBActRecTblDESCR**;**

TStringField **\***IBActRecTblActTypeLU**;**

TIntegerField **\***IBPrinterTblID**;**

TIntegerField **\***IBPrinterTblMAN\_ID**;**

TIBStringField **\***IBPrinterTblMODEL**;**

TIntegerField **\***IBPrinterTblPAGES\_PER\_MIN**;**

TDateField **\***IBPrinterTblINVOICE\_DATE**;**

TIntegerField **\***IBPrinterTblINVOICE\_NR**;**

TDateField **\***IBPrinterTblEXPIRE\_DATE**;**

TDateField **\***IBPrinterTblSCRAP\_DATE**;**

TIntegerField **\***IBPrinterTblOFFICE\_ID**;**

TStringField **\***IBPrinterTblManufactLU**;**

TStringField **\***IBPrinterTblOfficeLU**;**

TIntegerField **\***IBUpsTblID**;**

TIntegerField **\***IBUpsTblMAN\_ID**;**

TIBStringField **\***IBUpsTblMODEL**;**

TIntegerField **\***IBUpsTblMAX\_STATIONS**;**

TIntegerField **\***IBUpsTblMAX\_HOURS**;**

TDateField **\***IBUpsTblINVOICE\_DATE**;**

TIntegerField **\***IBUpsTblINVOICE\_NR**;**

TDateField **\***IBUpsTblEXPIRE\_DATE**;**

TDateField **\***IBUpsTblSCRAP\_DATE**;**

TIntegerField **\***IBUpsTblOFFICE\_ID**;**

TStringField **\***IBUpsTblManufactLU**;**

TStringField **\***IBUpsTblOfficeLU**;**

TIntegerField **\***IBSwitchTblID**;**

TIntegerField **\***IBSwitchTblMAN\_ID**;**

TIBStringField **\***IBSwitchTblMODEL**;**

TIntegerField **\***IBSwitchTblSPEED\_GBIT**;**

TIntegerField **\***IBSwitchTblPORT\_NUM**;**

TDateField **\***IBSwitchTblINVOICE\_DATE**;**

TIntegerField **\***IBSwitchTblINVOICE\_NR**;**

TDateField **\***IBSwitchTblEXPIRE\_DATE**;**

TDateField **\***IBSwitchTblSCRAP\_DATE**;**

TIntegerField **\***IBSwitchTblOFFICE\_ID**;**

TStringField **\***IBSwitchTblManufactLU**;**

TStringField **\***IBSwitchTblOfficeLU**;**

TIntegerField **\***IBCmpntTblID**;**

TIntegerField **\***IBCmpntTblCOMPUTER\_ID**;**

TIntegerField **\***IBCmpntTblCOMPONENT\_TYPE\_ID**;**

TIntegerField **\***IBCmpntTblCOMPONENT\_ID**;**

TDateField **\***IBCmpntTblEXPIRE\_DATE**;**

TDateField **\***IBCmpntTblINVOICE\_DATE**;**

TIntegerField **\***IBCmpntTblINVOICE\_NR**;**

TStringField **\***IBCmpntTblCmpntTypeLU**;**

TIntegerField **\***IBPersonTblID**;**

TIBStringField **\***IBPersonTblEGN**;**

TIBStringField **\***IBPersonTblFNAME**;**

TIBStringField **\***IBPersonTblSNAME**;**

TIBStringField **\***IBPersonTblLNAME**;**

TIBStringField **\***IBPersonTblPHONE**;**

TIBStringField **\***IBPersonTblEMAIL**;**

TIntegerField **\***IBPersonTblPOSITION\_ID**;**

TIntegerField **\***IBPersonTblDEPARTMENT\_ID**;**

TStringField **\***IBPersonTblPositionLU**;**

TStringField **\***IBPersonTblDepartmentLU**;**

TIntegerField **\***IBCpuTblnID**;**

TIntegerField **\***IBCpuTblnMAN\_ID**;**

TIBStringField **\***IBCpuTblnMODEL**;**

TIntegerField **\***IBCpuTblnCLOCK\_MHZ**;**

TIntegerField **\***IBCpuTblnCORE\_NUM**;**

TIntegerField **\***IBCpuTblnCACHE\_KB**;**

TStringField **\***IBCpuTblnManufactLU**;**

TIntegerField **\***IBHddTblnID**;**

TIntegerField **\***IBHddTblnMAN\_ID**;**

TIBStringField **\***IBHddTblnMODEL**;**

TFloatField **\***IBHddTblnFORMFCTR**;**

TIntegerField **\***IBHddTblnCAPACITY\_GB**;**

TIntegerField **\***IBHddTblnCACHE\_KB**;**

TIntegerField **\***IBHddTblnRPM**;**

TStringField **\***IBHddTblnManufactLU**;**

TIntegerField **\***IBLanTblnID**;**

TIntegerField **\***IBLanTblnMAN\_ID**;**

TIBStringField **\***IBLanTblnMODEL**;**

TIntegerField **\***IBLanTblnSPEED\_GBPITS**;**

TStringField **\***IBLanTblnManufactLU**;**

TIntegerField **\***IBMonitorTblnID**;**

TIntegerField **\***IBMonitorTblnMAN\_ID**;**

TIBStringField **\***IBMonitorTblnMODEL**;**

TFloatField **\***IBMonitorTblnSCREEN\_SIZE**;**

TStringField **\***IBMonitorTblnManufactLU**;**

TIntegerField **\***IBOpticalTblnID**;**

TIntegerField **\***IBOpticalTblnMAN\_ID**;**

TIBStringField **\***IBOpticalTblnMODEL**;**

TIntegerField **\***IBOpticalTblnWRITESPEED**;**

TIntegerField **\***IBOpticalTblnREADSPEED**;**

TIBStringField **\***IBOpticalTblnFORMATS**;**

TStringField **\***IBOpticalTblnManufactLU**;**

TIntegerField **\***IBOtherTblnID**;**

TIntegerField **\***IBOtherTblnMAN\_ID**;**

TIBStringField **\***IBOtherTblnDESCR**;**

TStringField **\***IBOtherTblnManufactLU**;**

TStringField **\***IBRamTblnManufactLU**;**

TIntegerField **\***IBRamTblnID**;**

TIntegerField **\***IBRamTblnMAN\_ID**;**

TIBStringField **\***IBRamTblnMODEL**;**

TIntegerField **\***IBRamTblnCAPACITY\_MB**;**

TIntegerField **\***IBRamTblnCLOCK\_MHZ**;**

TIntegerField **\***IBVgaTblnID**;**

TIntegerField **\***IBVgaTblnMAN\_ID**;**

TIBStringField **\***IBVgaTblnMODEL**;**

TIntegerField **\***IBVgaTblnCAPACITY\_MB**;**

TIntegerField **\***IBVgaTblnCPUCLOCK\_MHZ**;**

TIntegerField **\***IBVgaTblnCORE\_NUM**;**

TStringField **\***IBVgaTblnManufactLU**;**

TIntegerField **\***IBRouterTblID**;**

TIntegerField **\***IBRouterTblMAN\_ID**;**

TIBStringField **\***IBRouterTblMODEL**;**

TIntegerField **\***IBRouterTblMAX\_STATIONS**;**

TIntegerField **\***IBRouterTblRANGE\_METERS**;**

TDateField **\***IBRouterTblINVOICE\_DATE**;**

TIntegerField **\***IBRouterTblINVOICE\_NR**;**

TDateField **\***IBRouterTblEXPIRE\_DATE**;**

TDateField **\***IBRouterTblSCRAP\_DATE**;**

TIntegerField **\***IBRouterTblOFFICE\_ID**;**

TStringField **\***IBRouterTblManufactLU**;**

TStringField **\***IBRouterTblOfficeLU**;**

TIBQuery **\***ReferenceQRY**;**

TDataSource **\***ReferenceDS**;**

TIBStringField **\***IBCmpntTblSERIAL\_NR**;**

TIBQuery **\***ComponentQuery**;**

TDataSource **\***ComponentQueryDS**;**

TIntegerField **\***ComponentQueryID**;**

TIntegerField **\***ComponentQueryCOMPONENT\_ID**;**

TIntegerField **\***ComponentQueryCOMPONENT\_TYPE\_ID**;**

TIBStringField **\***ComponentQueryCOMPONENT\_TYPE**;**

TIBStringField **\***ComponentQueryCOMPONENT\_DESCR**;**

TIBStringField **\***ComponentQuerySERIAL\_NR**;**

TIntegerField **\***ComponentQueryINVOICE\_NR**;**

TDateField **\***ComponentQueryINVOICE\_DATE**;**

TDateField **\***ComponentQueryEXPIRE\_DATE**;**

TIntegerField **\***IBCmpntTypeTblnID**;**

TIBStringField **\***IBCmpntTypeTblnCOMPONENT\_TYPE**;**

TIntegerField **\***IBOfficeTblID**;**

TIntegerField **\***IBOfficeTblCITY\_ID**;**

TIBStringField **\***IBOfficeTblNAME**;**

TIBStringField **\***IBOfficeTblADDRESS**;**

TStringField **\***IBOfficeTblCityLU**;**

void \_\_fastcall IBActRecTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBActTypeTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCityTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCmpntTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCmpntTypeTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCmptrTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCmptrTypeTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCpuTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBDepartTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBHddTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBLanTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBSwitchTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBRouterTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBManufactTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBMonitorTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBMunicipTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBOfficeTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBOpticalTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBOtherTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBPersonTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBPositionTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBPrinterTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBRamTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBUpsTblBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBRegionTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBVgaTblnBeforePost**(**TDataSet **\***DataSet**);**

void \_\_fastcall IBCmptrTblAfterScroll**(**TDataSet **\***DataSet**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TIBDataModule**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TIBDataModule **\***IBDataModule**;**

//---------------------------------------------------------------------------

#endif

**DModFRM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "DModFRM.h"

#include "IBSysAdminMainFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TIBDataModule **\***IBDataModule**;**

//---------------------------------------------------------------------------

\_\_fastcall TIBDataModule**::**TIBDataModule**(**TComponent**\*** Owner**)**

**:** TDataModule**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TIBDataModule**::**IBActRecTblBeforePost**(**TDataSet **\***DataSet**)**

**{**

**if(**IBActRecTbl**->**State **==** dsInsert)

**{**

IBActRecIDSP**->**Prepare**();**

IBActRecIDSP**->**ExecProc**();**

IBActRecTbl**->**FieldByName**(**"ID"**)->**AsInteger **=**

IBActRecIDSP**->**ParamByName**(**"ID"**)->**AsInteger**;**

**}**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TIBDataModule**::**IBCmptrTblAfterScroll**(**TDataSet **\***DataSet**)**

**{**

**if(**ComponentQuery**->**Active)

ComponentQuery**->**Active **=false;**

ComponentQuery**->**Prepare**();**

ComponentQuery**->**Open**();**

**}**

//---------------------------------------------------------------------------

**IBSysAdminMainFRM.h**

#ifndef IBSysAdminMainFRMH

#define IBSysAdminMainFRMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <Menus.hpp>

#include <ComCtrls.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

#include <ExtCtrls.hpp>

#include <Buttons.hpp>

//---------------------------------------------------------------------------

#include <time.h>

#include <Dialogs.hpp>

#include <fstream.h>

//#include <ibase.h>

//---------------------------------------------------------------------------

class TForm1 **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TMainMenu **\***MainMenu**;**

TMenuItem **\***N1**;**

TMenuItem **\***N2**;**

TMenuItem **\***N3**;**

TMenuItem **\***N4**;**

TMenuItem **\***N5**;**

TMenuItem **\***N6**;**

TMenuItem **\***N7**;**

TMenuItem **\***N8**;**

TMenuItem **\***N9**;**

TMenuItem **\***N10**;**

TMenuItem **\***N11**;**

TMenuItem **\***N12**;**

TMenuItem **\***N13**;**

TMenuItem **\***N14**;**

TMenuItem **\***N15**;**

TMenuItem **\***N17**;**

TMenuItem **\***N18**;**

TMenuItem **\***N19**;**

TMenuItem **\***N20**;**

TMenuItem **\***U1**;**

TMenuItem **\***N21**;**

TMenuItem **\***N22**;**

TMenuItem **\***N23**;**

TMenuItem **\***N24**;**

TMenuItem **\***N25**;**

TMenuItem **\***N27**;**

TMenuItem **\***N16**;**

TMenuItem **\***N28**;**

TMenuItem **\***N30**;**

TMenuItem **\***N32**;**

TMenuItem **\***N26**;**

TMenuItem **\***N29**;**

TMenuItem **\***N31**;**

TMenuItem **\***N33**;**

TPageControl **\***PageControl**;**

TTabSheet **\***TabSheetComputers**;**

TSplitter **\***CmptrSplitter**;**

TDBGrid **\***IBCmptrTblDBGrid**;**

TDBGrid **\***IBCmpntDBGrid**;**

TGroupBox **\***CmptrTblGB**;**

TLabel **\***OrderLabelCmptr**;**

TLabel **\***SearchLabelCmptr**;**

TEdit **\***CmptrTblSearchEB**;**

TComboBox **\***OrderComboCmptr**;**

TComboBox **\***SearchComboCmptr**;**

TTabSheet **\***TabSheetPrinters**;**

TSplitter **\***PrinterSplitter**;**

TDBGrid **\***IBPrinterTblDBGrid**;**

TGroupBox **\***PrinterTblGB**;**

TLabel **\***OrderLabelPrinter**;**

TLabel **\***SearchLabelPrinter**;**

TEdit **\***PrinterTblSearchEB**;**

TComboBox **\***OrderComboPrinter**;**

TComboBox **\***SearchComboPrinter**;**

TTabSheet **\***TabSheetUPS**;**

TSplitter **\***UpsSplitter**;**

TDBGrid **\***IBUpsTblDBGrid**;**

TGroupBox **\***UpsTblGB**;**

TLabel **\***OrderLabelUps**;**

TLabel **\***SearchLabelUps**;**

TEdit **\***UpsTblSearchEB**;**

TComboBox **\***OrderComboUps**;**

TComboBox **\***SearchComboUps**;**

TTabSheet **\***TabSheetSwitches**;**

TSplitter **\***Splitter1**;**

TDBGrid **\***IBSwitchTblDBGrid**;**

TGroupBox **\***SwitchTblGB**;**

TLabel **\***OrderLabelSwitch**;**

TLabel **\***SearchLabelSwitch**;**

TEdit **\***SwitchTblSearchEB**;**

TComboBox **\***OrderComboSwitch**;**

TComboBox **\***SearchComboSwitch**;**

TTabSheet **\***TabSheetRouters**;**

TSplitter **\***RouterSplitter**;**

TDBGrid **\***RouterDBGrid**;**

TGroupBox **\***RouterTblGB**;**

TLabel **\***OrderLabelRouter**;**

TLabel **\***SearchLabelRouter**;**

TEdit **\***RouterTblSearchEB**;**

TComboBox **\***OrderComboRouter**;**

TComboBox **\***SearchComboRouter**;**

TMenuItem **\***N34**;**

TMenuItem **\***N35**;**

void \_\_fastcall IBCmptrTblDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBPrinterTblDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBSwitchTblDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBUpsTblDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall CmptrTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall RouterDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall N7Click**(**TObject **\***Sender**);**

void \_\_fastcall N15Click**(**TObject **\***Sender**);**

void \_\_fastcall OrderComboCmptrSelect**(**TObject **\***Sender**);**

void \_\_fastcall OrderComboPrinterSelect**(**TObject **\***Sender**);**

void \_\_fastcall PrinterTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall OrderComboUpsSelect**(**TObject **\***Sender**);**

void \_\_fastcall UpsTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall OrderComboSwitchSelect**(**TObject **\***Sender**);**

void \_\_fastcall SwitchTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall OrderComboRouterSelect**(**TObject **\***Sender**);**

void \_\_fastcall RouterTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall N28Click**(**TObject **\***Sender**);**

void \_\_fastcall N27Click**(**TObject **\***Sender**);**

void \_\_fastcall N11Click**(**TObject **\***Sender**);**

void \_\_fastcall N9Click**(**TObject **\***Sender**);**

void \_\_fastcall N23Click**(**TObject **\***Sender**);**

void \_\_fastcall N24Click**(**TObject **\***Sender**);**

void \_\_fastcall N25Click**(**TObject **\***Sender**);**

void \_\_fastcall N13Click**(**TObject **\***Sender**);**

void \_\_fastcall PageControlChanging**(**TObject **\***Sender**,**bool**&**AllowChange**);**

void \_\_fastcall PageControlChange**(**TObject **\***Sender**);**

void \_\_fastcall N34Click**(**TObject **\***Sender**);**

void \_\_fastcall N35Click**(**TObject **\***Sender**);**

void \_\_fastcall N30Click**(**TObject **\***Sender**);**

void \_\_fastcall N29Click**(**TObject **\***Sender**);**

void \_\_fastcall N33Click**(**TObject **\***Sender**);**

void \_\_fastcall N32Click**(**TObject **\***Sender**);**

void \_\_fastcall N2Click**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

bool OnlyDigits**(**String str)**;**

void WriteReferenceToFile**(**String)**;**

\_\_fastcall TForm1**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TForm1 **\***Form1**;**

//---------------------------------------------------------------------------

#endif

**IBSysAdminMainFRM.cpp**

#include <vcl.h>

#include <cstdio.h>

#pragma hdrstop

#include "IBSysAdminMainFRM.h"

#include "DModFRM.h"

#include "ManufactInputFORM.h"

#include "ComponentEditFORM.h"

#include "PersonsFORM.h"

#include "RefFORM.h"

#include "ActRecFORM.h"

#include "ManufactFORM.h"

#include "DepFORM.h"

#include "PosFORM.h"

#include "OfficeFORM.h"

#include "ActTypeFORM.h"

//-----------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 **\***Form1**;**

//-----------------------------------------

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <iostream>

#include <iomanip>

#include <string>

//-----------------------------------------

bool TForm1**::**OnlyDigits**(**String str)

**{**

char**\*** str\_copy **=** str**.**c\_str**();**

**for(**int i **=**0**;** str\_copy**[** i **]!=**'\0'**;** i**++**)

**if(!**isdigit**(**str\_copy**[** i **]**))

**returnfalse;**

**returntrue;**

**}**

//-----------------------------------------

\_\_fastcall TForm1**::**TForm1**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**IBCmptrTblDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBCmptrTbl**->**State **==** dsInsert**)**

**||** (IBDataModule**->**IBCmptrTbl**->**State **==** dsEdit))

IBDataModule**->**IBCmptrTbl**->**Post**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**CmptrTblSearchEBChange**(**TObject **\***Sender**)**

**{**

AnsiString FieldName **=**""**;**

TLocateOptions Opts**;**

Opts**.**Clear**();**

Opts **<<** loPartialKey**;**

Opts **<<** loCaseInsensitive**;**

**switch(**SearchComboCmptr**->**ItemIndex**)**

**{**

**case**0**:** FieldName **=**"MODEL"**;break;**

**case**1**:** FieldName **=**"INVOICE\_NR"**;break;**

**case**2**:** FieldName **=**"INTERFACES\_DESCR"**;break;**

**default:** FieldName **=**"MODEL"**;**

**}**

IBDataModule**->**IBCmptrTbl**->**Locate**(**FieldName**,** CmptrTblSearchEB**->**Text**,** Opts)**;**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N7Click**(**TObject **\***Sender**)**

**{**

PageControl**->**ActivePage **=** PageControl**->**Pages**[static\_cast<**TMenuItem**\*>(**Sender**)->**Tag**];**

//PageControl->OnChanging; //not working?????

//PageControl->OnChange;

N18**->**Enabled **=false;**

N15**->**Enabled **=false;**

N19**->**Enabled **=false;**

N26**->**Enabled **=false;**

N31**->**Enabled **=false;**

U1**->**Enabled **=false;**

**switch(**PageControl**->**TabIndex)

**{**

**case**0**:** N18**->**Enabled **=true;**

N15**->**Enabled **=true;**

**break;**

**case**1**:** N19**->**Enabled **=true;break;**

**case**2**:** U1**->**Enabled **=true;break;**

**case**3**:** N31**->**Enabled **=true;break;**

**case**4**:** N26**->**Enabled **=true;break;**

**}**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N15Click**(**TObject **\***Sender**)**

**{**

**if(**ComponentEditFRM**->**ShowModal**()!=**mrOk**)**

**return;**

**else**

**{**

IBDataModule**->**IBCmpntTbl**->**Append**();**

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_TYPE\_ID"**)->**AsInteger **=**

ComponentEditFRM**->**IBCmpntTypeTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPUTER\_ID"**)->**AsInteger **=**

IBDataModule**->**IBCmptrTbl**->**FieldByName**(**"ID"**)->**AsInteger**;**

**if(!**ComponentEditFRM**->**CmpntSerialEB**->**Text**.**IsEmpty**()**)

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"SERIAL\_NR"**)->**AsString **=**

ComponentEditFRM**->**CmpntSerialEB**->**Text**.**SetLength**(**39)**.**c\_str**();**

**if(**OnlyDigits**(**ComponentEditFRM**->**CmpntInvNrEB**->**Text)

**&&!**ComponentEditFRM**->**CmpntInvNrEB**->**Text**.**IsEmpty**()**)

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"INVOICE\_NR"**)->**AsInteger **=**

StrToInt**(**ComponentEditFRM**->**CmpntInvNrEB**->**Text**.**c\_str**()**)**;**

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"INVOICE\_DATE"**)->**AsString **=**

ComponentEditFRM**->**CmpntInvDateDTPicker**->**Date**.**DateString**().**SubString**(**0**,**10**).**c\_str**();**

IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"EXPIRE\_DATE"**)->**AsString **=**

ComponentEditFRM**->**CmpntExpDateDTPicker**->**Date**.**DateString**().**SubString**(**0**,**10**).**c\_str**();**

**switch(**ComponentEditFRM**->**IBCmpntTypeTbln**->**FieldByName**(**"ID"**)->**AsInteger)

**{**

**case**1**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBVgaTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**2**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBLanTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**3**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBCpuTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**4**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBHddTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**5**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBMonitorTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**6**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBOpticalTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**7**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBOtherTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**case**8**:** IBDataModule**->**IBCmpntTbl**->**FieldByName**(**"COMPONENT\_ID"**)->**AsInteger **=**

IBDataModule**->**IBRamTbln**->**FieldByName**(**"ID"**)->**AsInteger**;**

**break;**

**}**

IBDataModule**->**IBCmpntTbl**->**Post**();**

**if(**IBDataModule**->**ComponentQuery**->**Active)

IBDataModule**->**ComponentQuery**->**Active **=false;**

IBDataModule**->**ComponentQuery**->**Prepare**();**

IBDataModule**->**ComponentQuery**->**Open**();**

ComponentEditFRM**->**CmpntSerialEB**->**Clear**();**

ComponentEditFRM**->**CmpntInvNrEB**->**Clear**();**

**}**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**OrderComboCmptrSelect**(**TObject **\***Sender**)**

**{**

**switch(**OrderComboCmptr**->**ItemIndex**)**

**{**

**case**0**:** IBDataModule**->**IBCmptrTbl**->**IndexName **=**"CmptrTblModelNDX"**;break;**

**case**1**:** IBDataModule**->**IBCmptrTbl**->**IndexName **=**"CmptrTblInvDateNDX"**;break;**

**case**2**:** IBDataModule**->**IBCmptrTbl**->**IndexName **=**"CmptrTblExpDateNDX"**;break;**

**case**3**:** IBDataModule**->**IBCmptrTbl**->**IndexName **=**"CmptrTblScrapNDX"**;break;**

**case**4**:** IBDataModule**->**IBCmptrTbl**->**IndexName **=**"CmptrTblInvNrNDX"**;break;**

**}**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N28Click**(**TObject **\***Sender**)**

**{**

TLocateOptions Opts**;**

Opts**.**Clear**();**

**if(**IBDataModule**->**IBCmpntTbl**->**Locate**(**"ID"**,**

IBDataModule**->**ComponentQuery**->**FieldByName**(**"ID"**)->**AsInteger**,** Opts))

**{**

IBDataModule**->**IBCmpntTbl**->**Delete**();**

**if(**IBDataModule**->**ComponentQuery**->**Active)

IBDataModule**->**ComponentQuery**->**Active **=false;**

IBDataModule**->**ComponentQuery**->**Prepare**();**

IBDataModule**->**ComponentQuery**->**Open**();**

**}**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N27Click**(**TObject **\***Sender**)**

**{**

IBDataModule**->**IBCmpntTbl**->**First**();**

**if(**IBDataModule**->**IBCmpntTbl**->**Bof)

**{**

**while(!**IBDataModule**->**IBCmpntTbl**->**Eof)

IBDataModule**->**IBCmpntTbl**->**Delete**();**

**}**

IBDataModule**->**IBCmptrTbl**->**Delete**();**

IBDataModule**->**IBCmptrTbl**->**Refresh**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N11Click**(**TObject **\***Sender**)**

**{**

PersonsFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N9Click**(**TObject **\***Sender**)**

**{**

ActRecFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N23Click**(**TObject **\***Sender**)**

**{**

ManufactFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N24Click**(**TObject **\***Sender**)**

**{**

DepFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N25Click**(**TObject **\***Sender**)**

**{**

PosFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N13Click**(**TObject **\***Sender**)**

**{**

Form1**->**Close**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**PageControlChanging**(**TObject **\***Sender**,**

bool**&**AllowChange**)**

**{**

N18**->**Enabled **=false;**

N15**->**Enabled **=false;**

N19**->**Enabled **=false;**

N26**->**Enabled **=false;**

N31**->**Enabled **=false;**

U1**->**Enabled **=false;**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**PageControlChange**(**TObject **\***Sender**)**

**{**

**switch(**PageControl**->**TabIndex)

**{**

**case**0**:** N18**->**Enabled **=true;**

N15**->**Enabled **=true;**

**break;**

**case**1**:** N19**->**Enabled **=true;break;**

**case**2**:** U1**->**Enabled **=true;break;**

**case**3**:** N31**->**Enabled **=true;break;**

**case**4**:** N26**->**Enabled **=true;break;**

**}**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N34Click**(**TObject **\***Sender**)**

**{**

OfficeFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N35Click**(**TObject **\***Sender**)**

**{**

ActTypeFRM**->**ShowModal**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N30Click**(**TObject **\***Sender**)**

**{**

IBDataModule**->**IBPrinterTbl**->**Delete**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N29Click**(**TObject **\***Sender**)**

**{**

IBDataModule**->**IBRouterTbl**->**Delete**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N33Click**(**TObject **\***Sender**)**

**{**

IBDataModule**->**IBSwitchTbl**->**Delete**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N32Click**(**TObject **\***Sender**)**

**{**

IBDataModule**->**IBUpsTbl**->**Delete**();**

**}**

//-----------------------------------------

void \_\_fastcall TForm1**::**N2Click**(**TObject **\***Sender**)**

**{**

RefFRM**->**ShowModal**();**

**}**

//-----------------------------------------

**ComponentEditFORM.h**

#ifndef ComponentEditFORMH

#define ComponentEditFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <ComCtrls.hpp>

#include <DB.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

#include <IBCustomDataSet.hpp>

#include <IBTable.hpp>

#include <Buttons.hpp>

#include <ExtCtrls.hpp>

#include <DBCtrls.hpp>

//---------------------------------------------------------------------------

class TComponentEditFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TLabel **\***CmpntTypeLBL**;**

TLabel **\***CmpntSerialLBL**;**

TLabel **\***CmpntInvNrLBL**;**

TLabel **\***CmpntInvDateLBL**;**

TLabel **\***CmpntExpDateLBL**;**

TDBGrid **\***ComponentTypeDBGrid**;**

TEdit **\***CmpntSerialEB**;**

TEdit **\***CmpntInvNrEB**;**

TDateTimePicker **\***CmpntInvDateDTPicker**;**

TDateTimePicker **\***CmpntExpDateDTPicker**;**

TPageControl **\***ComponentPages**;**

TTabSheet **\***TabSheet1**;**

TTabSheet **\***TabSheet2**;**

TTabSheet **\***TabSheet3**;**

TTabSheet **\***TabSheet4**;**

TTabSheet **\***TabSheet5**;**

TTabSheet **\***TabSheet6**;**

TTabSheet **\***TabSheet7**;**

TTabSheet **\***TabSheet8**;**

TDBGrid **\***IBVgaDBGrid**;**

TDBGrid **\***IBLanDBGrid**;**

TDBGrid **\***IBCpuDBGrid**;**

TDBGrid **\***DBGrid7**;**

TIBTable **\***IBCmpntTypeTbln**;**

TDataSource **\***IBCmpntTypeDS**;**

TDBGrid **\***IBHddDBGrid**;**

TDBGrid **\***IBMonitorDBGrid**;**

TDBGrid **\***IBOpticalDBGrid**;**

TDBGrid **\***IBOtherDBGrid**;**

TDBGrid **\***IBRamDBGrid**;**

TIntegerField **\***IBCmpntTypeTblnID**;**

TIBStringField **\***IBCmpntTypeTblnCOMPONENT\_TYPE**;**

TBevel **\***Bevel**;**

TBitBtn **\***OKBitBtn**;**

TBitBtn **\***CancelBitBtn**;**

TDBNavigator **\***ComponentDBNavigator**;**

void \_\_fastcall IBCmpntTypeTblnAfterScroll**(**TDataSet **\***DataSet**);**

void \_\_fastcall FormCreate**(**TObject **\***Sender**);**

void \_\_fastcall IBVgaDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBLanDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall ComponentPagesExit**(**TObject **\***Sender**);**

void \_\_fastcall IBHddDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBMonitorDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBOpticalDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBOtherDBGridExit**(**TObject **\***Sender**);**

void \_\_fastcall IBRamDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TComponentEditFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TComponentEditFRM **\***ComponentEditFRM**;**

//---------------------------------------------------------------------------

#endif

**ComponentEditFORM.cpp**

//---------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "ComponentEditFORM.h"

#include "DModFRM.h"

//---------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TComponentEditFRM **\***ComponentEditFRM**;**

//---------------------------------------

\_\_fastcall TComponentEditFRM**::**TComponentEditFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------

void \_\_fastcall TComponentEditFRM**::**IBCmpntTypeTblnAfterScroll**(**

TDataSet **\***DataSet**)**

**{**

.ComponentPages**->**ActivePage **=**

ComponentPages**->**Pages**[**IBCmpntTypeTbln**->**FieldByName**(**"ID"**)->**AsInteger**-**1**];**

**switch(**IBCmpntTypeTbln**->**FieldByName**(**"ID"**)->**AsInteger**-**1)

**{**

**case**0**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBVgaDS**;break;**

**case**1**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBLanDS**;break;**

**case**2**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBCpuDS**;break;**

**case**3**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBHddDS**;break;**

**case**4**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBMonitorDS**;break;**

**case**5**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBOpticalDS**;break;**

**case**6**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBOtherDS**;break;**

**case**7**:** ComponentDBNavigator**->**DataSource **=** IBDataModule**->**IBRamDS**;break;**

**}**

**}**

//---------------------------------------

void \_\_fastcall TComponentEditFRM**::**FormCreate**(**TObject **\***Sender**)**

**{**

**for(**int i**=**0**;** i **<** ComponentPages**->**PageCount**;** i**++)**

ComponentPages**->**Pages**[**i**]->**TabVisible **=false;**

ComponentPages**->**ActivePage **=**

ComponentPages**->**Pages**[**IBCmpntTypeTbln**->**FieldByName**(**"ID"**)->**AsInteger**-**1**];**

**}**

//---------------------------------------

void \_\_fastcall TComponentEditFRM**::**IBVgaDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBVgaTbln**->**State **==** dsEdit**)||**

(IBDataModule**->**IBVgaTbln**->**State **==** dsInsert))

IBDataModule**->**IBVgaTbln**->**Post**();**

**}**

**RefFORM.h**

#ifndef RefFORMH

#define RefFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <Buttons.hpp>

#include <ComCtrls.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

#include <Dialogs.hpp>

//---------------------------------------------------------------------------

class TRefFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TGroupBox **\***ReferenceButtonsGB**;**

TLabel **\***QuerySaveLBL**;**

TBitBtn **\***ReferenceBitBtn**;**

TEdit **\***QuerySaveEB**;**

TBitBtn **\***QuerySaveBitBtn**;**

TGroupBox **\***ReferenceMakerGB**;**

TComboBox **\***MainCombo**;**

TComboBox **\***DeviceCombo**;**

TComboBox **\***ComponentCombo**;**

TGroupBox **\***PersonInfoGB**;**

TCheckBox **\***PersonInfoCB\_1**;**

TCheckBox **\***PersonInfoCB\_2**;**

TCheckBox **\***PersonInfoCB\_3**;**

TEdit **\***PersonInfoEB\_1**;**

TEdit **\***PersonInfoEB\_2**;**

TEdit **\***PersonInfoEB\_3**;**

TGroupBox **\***ActivityGB**;**

TLabel **\***ActivityLBL\_1**;**

TLabel **\***ActivityLBL\_2**;**

TEdit **\***ActivityEB**;**

TCheckBox **\***ActivityCB\_1**;**

TCheckBox **\***ActivityCB\_2**;**

TCheckBox **\***ActivityCB\_3**;**

TComboBox **\***ActivityCombo**;**

TDateTimePicker **\***ActivityDTPicker\_1**;**

TDateTimePicker **\***ActivityDTPicker\_2**;**

TGroupBox **\***CommonGB**;**

TLabel **\***CommonLabel\_1**;**

TLabel **\***CommonLabel\_2**;**

TLabel **\***CommonLabel\_3**;**

TLabel **\***CommonLabel\_4**;**

TLabel **\***Label1**;**

TLabel **\***Label2**;**

TLabel **\***Label3**;**

TLabel **\***Label4**;**

TCheckBox **\***CommonCB\_1**;**

TCheckBox **\***CommonCB\_2**;**

TCheckBox **\***CommonCB\_3**;**

TCheckBox **\***CommonCB\_4**;**

TCheckBox **\***CommonCB\_5**;**

TDateTimePicker **\***CommonDTPicker\_1**;**

TDateTimePicker **\***CommonDTPicker\_4**;**

TDateTimePicker **\***CommonDTPicker\_6**;**

TDateTimePicker **\***CommonDTPicker\_2**;**

TDateTimePicker **\***CommonDTPicker\_3**;**

TDateTimePicker **\***CommonDTPicker\_5**;**

TEdit **\***CommonEB\_2**;**

TEdit **\***CommonEB\_1**;**

TCheckBox **\***CommonCB\_6**;**

TGroupBox **\***MoreDevDetailsGB**;**

TLabel **\***CmpntDetLabel**;**

TLabel **\***CmpDetLBL\_1**;**

TLabel **\***CmpDetLBL\_2**;**

TLabel **\***Label5**;**

TLabel **\***Label6**;**

TLabel **\***Label7**;**

TLabel **\***Label8**;**

TEdit **\***CmpntDetEB\_1**;**

TEdit **\***CmpntDetEB\_2**;**

TEdit **\***CmpntDetEB\_3**;**

TEdit **\***CmpntDetEB\_4**;**

TEdit **\***CmpntDetEB\_5**;**

TEdit **\***CmpntDetEB\_6**;**

TEdit **\***CmpntDetEB\_7**;**

TCheckBox **\***CmpntDetCB\_1**;**

TCheckBox **\***CmpntDetCB\_2**;**

TCheckBox **\***CmpntDetCB\_3**;**

TCheckBox **\***CmpntDetCB\_4**;**

TDBGrid **\***ReferenceDBGrid**;**

TGroupBox **\***OfficeGB**;**

TEdit **\***OfficeEB\_1**;**

TCheckBox **\***OfficeCB\_1**;**

TCheckBox **\***OfficeCB\_2**;**

TEdit **\***OfficeEB\_2**;**

TEdit **\***CommonEB\_3**;**

TLabel **\***CommonLabel\_5**;**

TCheckBox **\***CommonCB\_7**;**

TSaveDialog **\***QuerySaveDialog**;**

TDBGrid **\***ManufactDBGrid**;**

void \_\_fastcall MainComboSelect**(**TObject **\***Sender**);**

void \_\_fastcall DeviceComboSelect**(**TObject **\***Sender**);**

void \_\_fastcall ComponentComboSelect**(**TObject **\***Sender**);**

void \_\_fastcall ReferenceBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall QuerySaveBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall FormClose**(**TObject **\***Sender**,** TCloseAction **&**Action**);**

private**:**// User declarations

public**:**// User declarations

void SetDBGridCaptions**();**

void WriteReferenceToFile**(**String s)**;**

\_\_fastcall TRefFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TRefFRM **\***RefFRM**;**

//---------------------------------------------------------------------------

#endif

**RefFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "RefFORM.h"

#include "DModFRM.h"

#include "IBSysAdminMainFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TRefFRM **\***RefFRM**;**

//---------------------------------------------------------------------------

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <iostream>

#include <iomanip>

#include <string>

//---------------------------------------------------------------------------

\_\_fastcall TRefFRM**::**TRefFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void TRefFRM**::**WriteReferenceToFile**(**String s)

**{**

fstream file**(**s**.**c\_str**(),** ios**::**out)**;**

**if(!**file**.**is\_open**()**)

**{**

**if(**MessageDlgPos**(**"Файлът не е създаден! Проверете дали имате\

необходимите права за създаване на файл"**,**

mtError**,** TMsgDlgButtons**()<<** mbRetry

**<<** mbCancel**,**0**,**

Form1**->**ClientOrigin**.**x **+** Form1**->**ClientWidth **/**3**,**

Form1**->**ClientOrigin**.**y **+** Form1**->**ClientHeight **/**3)

**==** mbRetry)

RefFRM**->**QuerySaveBitBtn**->**OnClick**;**

**}**

**else**

**{**

int l**;**

int col\_width **=**10**;**

int field\_count **=** IBDataModule**->**ReferenceQRY**->**FieldCount**;**

String s**;**

**if(!**RefFRM**->**QuerySaveEB**->**Text**.**IsEmpty**()**

**&&** Form1**->**OnlyDigits**(**RefFRM**->**QuerySaveEB**->**Text))

col\_width **=** StrToInt**(**RefFRM**->**QuerySaveEB**->**Text)**;**

**for(**int i **=**0**;** i **<** field\_count**;** i**++**)

**{**

s **=** RefFRM**->**ReferenceDBGrid**->**Columns**->operator[](**i**)->**Title**->**Caption**.**c\_str**();**

s **=** s**.**SetLength**(**col\_width)**.**c\_str**();**

l **=** s**.**Length**();**

file **<<**"|"**<<** s**.**c\_str**()<<** setw**(**col\_width **+**1**-** l)**;**

**}**

file **<<**"|"**<<** endl**;**

**for(**int i **=**0**;** i **<** col\_width **\*** field\_count **+** field\_count**;** i**++**)

file **<<**"-"**;**

file **<<** endl**;**

**if(**IBDataModule**->**ReferenceQRY**->**RecordCount **>**0)

IBDataModule**->**ReferenceQRY**->**First**();**

**while(!**IBDataModule**->**ReferenceQRY**->**Eof)

**{**

**for(**int i **=**0**;** i **<** field\_count**;** i**++**)

**{**

**if(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**DataType **==** ftString)

**{**

l **=** IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**AsString**.**Length**();**

file **<<**"|"**<<** IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)**

**->**AsString**.**SetLength**(**col\_width**).**c\_str**()**

**<<** setw**(**col\_width **+**1**-** l)**;**

**}**

**elseif(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**DataType **==** ftFloat)

**{**

s **=** FloatToStr**(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**AsFloat)**;**

s **=** s**.**SetLength**(**col\_width)**.**c\_str**();**

l **=** s**.**Length**();**

file **<<**"|"**<<** s**.**c\_str**()<<** setw**(**col\_width **+**1**-** l)**;**

**}**

**elseif(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**DataType**==**ftInteger)

**{**

s **=** IntToStr**(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**AsInteger)**;**

s **=** s**.**SetLength**(**col\_width)**.**c\_str**();**

l **=** s**.**Length**();**

file **<<**"|"**<<** s**.**c\_str**()<<** setw**(**col\_width **+**1**-** l)**;**

**}**

**elseif(**IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)->**DataType **==** ftDate)

**{**

s **=** IBDataModule**->**ReferenceQRY**->**Fields**->operator[](**i**)**

**->**AsDateTime**.**DateString**().**c\_str**();**

s **=** s**.**SetLength**(**10)**.**c\_str**();**//02.05.2016

l **=** s**.**Length**();**

file **<<**"|"**<<** s**.**c\_str**()<<** setw**(**col\_width **+**1**-** l)**;**

**}**

**}**

file **<<** setw**(**col\_width **+**1**-** l**)<<**"|"**<<** endl**;**

**for(**int i **=**0**;** i **<** col\_width **\*** field\_count **+** field\_count**;** i**++**)

file **<<**"-"**;**

file **<<** endl**;**

IBDataModule**->**ReferenceQRY**->**Next**();**

**}**

file**.**close**();**

QuerySaveEB**->**Clear**();**

**}**

**}**

//-------------------------------------------------------------------------------------------

void TRefFRM**::**SetDBGridCaptions**()**

**{**

**switch(**MainCombo**->**ItemIndex)

**{**

**case**0**:**//Device

**{**

ReferenceDBGrid**->**Columns**->operator[](**0**)->**Title**->**Caption **=**"Производител"**;**

ReferenceDBGrid**->**Columns**->operator[](**1**)->**Title**->**Caption **=**"Модел"**;**

ReferenceDBGrid**->**Columns**->operator[](**4**)->**Title**->**Caption **=**"Офис"**;**

ReferenceDBGrid**->**Columns**->operator[](**5**)->**Title**->**Caption **=**"Дата на фактура"**;**

ReferenceDBGrid**->**Columns**->operator[](**6**)->**Title**->**Caption **=**"Номер на фактура"**;**

ReferenceDBGrid**->**Columns**->operator[](**7**)->**Title**->**Caption **=**"Дата на гаранция"**;**

//------------FUNCTION IS CUT HERE!

void \_\_fastcall TRefFRM**::**MainComboSelect**(**TObject **\***Sender**)**

**{**

ActivityGB**->**Visible **=false;**

PersonInfoGB**->**Visible **=false;**

CommonGB**->**Visible **=false;**

CommonEB\_3**->**Enabled **=false;**

CommonCB\_7**->**Enabled **=false;**

CommonCB\_7**->**Checked **=false;**

OfficeGB**->**Visible **=false;**

MoreDevDetailsGB**->**Visible **=false;**

ComponentCombo**->**ItemIndex **=-**1**;**

ComponentCombo**->**Enabled **=false;**

DeviceCombo**->**ItemIndex **=-**1**;**

DeviceCombo**->**Enabled **=false;**

**switch(**MainCombo**->**ItemIndex)

**{**

**case**0**:** DeviceCombo**->**Enabled **=true;**

**break;**

//------------FUNCTION IS CUT HERE!

void \_\_fastcall TRefFRM**::**DeviceComboSelect**(**TObject **\***Sender**)**

**{**

**switch(**DeviceCombo**->**ItemIndex)

**{**

**case**0**:** CommonGB**->**Visible **=true;**

ComponentCombo**->**Enabled **=true;**

**break;**

**case**1**:** MoreDevDetailsGB**->**Caption **=**"Детайли на принтер:"**;**

CmpntDetCB\_1**->**Enabled **=true;**

CmpntDetCB\_1**->**Caption **=**"Стр/Мин"**;**

CommonEB\_3**->**Enabled **=false;**

CommonCB\_7**->**Enabled **=false;**

CommonCB\_7**->**Checked **=false;**

CommonGB**->**Visible **=true;**

MoreDevDetailsGB**->**Visible **=true;**

**break;**

//------------FUNCTION IS CUT HERE!

void \_\_fastcall TRefFRM**::**ComponentComboSelect**(**TObject **\***Sender**)**

**{**

**switch(**ComponentCombo**->**ItemIndex)

**{**

**case**0**:** CmpntDetCB\_1**->**Enabled **=true;**

CmpntDetCB\_1**->**Caption **=**"Тактова честота(MHz)"**;**

CmpntDetCB\_2**->**Enabled **=true;**

CmpntDetCB\_2**->**Caption **=**"Брой ядра"**;**

CmpntDetCB\_3**->**Enabled **=true;**

CmpntDetCB\_3**->**Caption **=**"Кеш памет(KB)"**;**

MoreDevDetailsGB**->**Caption **=**"Детайли на процесор"**;**

CommonEB\_3**->**Enabled **=true;**

CommonCB\_7**->**Enabled **=true;**

MoreDevDetailsGB**->**Visible **=true;**

**break;**

//------------FUNCTION IS CUT HERE!

void \_\_fastcall TRefFRM**::**ReferenceBitBtnClick**(**TObject **\***Sender**)**

**{**

String SelectClause **=**"SELECT "**;**

String FromClause **=**"\nFROM "**;**

String WhereClause **=**"\nWHERE "**;**

String FullQuery**;**

bool execute\_flag **=true;**

**if(**IBDataModule**->**ReferenceQRY**->**Active)

IBDataModule**->**ReferenceQRY**->**Active **=false;**

**switch(**MainCombo**->**ItemIndex)

**{**

**case**0**:**//Device

**switch(**DeviceCombo**->**ItemIndex)

**{**

**case**0**:**//Computer

**{**

FromClause **+=**

"COMPUTER\_TBL CR\

\nJOIN COMPONENT\_TBL CT ON CR.ID = CT.COMPUTER\_ID\

\nJOIN OFFICE\_TBL O ON CR.OFFICE\_ID = O.ID\

\nJOIN PERSON\_TBL PS ON CR.PERSON\_ID = PS.ID "**;**

SelectClause **+=**"CR.MODEL AS CMPTR\_MODEL,\

\nCR.SCRAP\_DATE AS CMPTR\_SCRAP\_DATE,\

\nMAN.NAME AS CMPNT\_MANUFACT,\

\nCT.INVOICE\_DATE AS CMPNT\_INV\_DATE,\

\nCT.EXPIRE\_DATE AS CMPNT\_EXP\_DATE,\

\nCT.INVOICE\_NR AS CMPNT\_INV\_NR,\

\nO.NAME AS OFF\_NAME, PS.FNAME, PS.LNAME"**;**

**switch(**ComponentCombo**->**ItemIndex)

**{**

**case**0**:**//CPU

SelectClause **+=**", CP.MODEL, CP.CLOCK\_MHZ,\

CP.CORE\_NUM, CP.CACHE\_KB "**;**

FromClause **+=**"JOIN CPU\_TBLN CP\

ON CP.ID = CT.COMPONENT\_ID\

JOIN MANUFACTURER\_TBLN MAN ON MAN.ID = CP.MAN\_ID"**;**

WhereClause **+=**"CT.COMPONENT\_TYPE\_ID = 3 "**;**

//--------Concrete device search options------------------------

**if(**CmpntDetCB\_1**->**Checked

**&&** CmpntDetCB\_1**->**Enabled

**&&!**CmpntDetEB\_1**->**Text**.**IsEmpty**()**

**&&** CmpntDetEB\_2**->**Text**.**Length**()>**0

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_1**->**Text)

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_2**->**Text))

WhereClause **+=**

"\nAND CP.CLOCK\_MHZ BETWEEN "**+** CmpntDetEB\_1**->**Text

**+**" AND "**+**CmpntDetEB\_2**->**Text **+**" "**;**

**if(**CmpntDetCB\_2**->**Checked

**&&** CmpntDetCB\_2**->**Enabled

**&&** CmpntDetEB\_3**->**Text**.**Length**()>**0

**&&** CmpntDetEB\_4**->**Text**.**Length**()>**0

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_3**->**Text)

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_4**->**Text))

WhereClause **+=**

"\nAND CP.CORE\_NUM BETWEEN "**+** CmpntDetEB\_3**->**Text

**+**" AND "**+** CmpntDetEB\_4**->**Text **+**" "**;**

**if(**CmpntDetCB\_3**->**Checked

**&&** CmpntDetCB\_3**->**Enabled

**&&** CmpntDetEB\_5**->**Text**.**Length**()>**0

**&&** CmpntDetEB\_6**->**Text**.**Length**()>**0

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_5**->**Text)

**&&** Form1**->**OnlyDigits**(**CmpntDetEB\_6**->**Text))

WhereClause **+=**

"\nAND CP.CACHE\_KB BETWEEN "**+** CmpntDetEB\_5**->**Text

**+**" AND "**+** CmpntDetEB\_6**->**Text **+**" "**;**

**if(**CommonCB\_7**->**Checked

**&&** CommonCB\_7**->**Enabled

**&&** CommonEB\_3**->**Text**.**Length**()>**0)

WhereClause **+=**

"\nAND CT.SERIAL\_NR LIKE '%"**+** CommonEB\_3**->**Text **+**"%' "**;**

//--------Common device search options------------------------

**if(**CommonCB\_1**->**Checked **&&** CommonCB\_1**->**Enabled)

WhereClause **+=**

"\nAND CR.SCRAP\_DATE BETWEEN CAST('"

**+** CommonDTPicker\_1**->**Date**.**DateString**().**SubString**(**0**,**10**)**

**+**"' AS DATE) AND CAST('"

**+** CommonDTPicker\_2**->**Date**.**DateString**().**SubString**(**0**,**10**)**

**+**"' AS DATE) "**;**

**if(**CommonCB\_2**->**Checked **&&** CommonCB\_2**->**Enabled)

WhereClause **+=**

"\nAND CT.INVOICE\_DATE BETWEEN CAST('"

**+** CommonDTPicker\_4**->**Date**.**DateString**().**SubString**(**0**,**10**)**

**+**"' AS DATE) AND CAST('"

**+** CommonDTPicker\_3**->**Date**.**DateString**().**SubString**(**0**,**10**)+**"' AS DATE) "**;**

**if(**CommonCB\_3**->**Checked **&&** CommonCB\_3**->**Enabled)

WhereClause **+=**

"\nAND CT.EXPIRE\_DATE BETWEEN CAST('"

**+** CommonDTPicker\_6**->**Date**.**DateString**().**SubString**(**0**,**10**)**

**+**"' AS DATE) AND CAST('"

**+** CommonDTPicker\_5**->**Date**.**DateString**().**SubString**(**0**,**10**)**

**+**"' AS DATE) "**;**

**if(**CommonCB\_4**->**Checked **&&** CommonCB\_4**->**Enabled)

WhereClause **+=**"\nAND CP.MAN\_ID = "**+** IntToStr**(**

IBDataModule**->**IBManufactTbln**->**FieldByName**(**"ID"**)->**AsInteger)**+**" "**;**

**if(**CommonCB\_5**->**Checked **&&** CommonCB\_5**->**Enabled

**&&** CommonEB\_1**->**Text**.**Length**()>**0

**&&** Form1**->**OnlyDigits**(**CommonEB\_1**->**Text**)**)

WhereClause **+=**"\nAND CT.INVOICE\_NR = "**+** CommonEB\_1**->**Text **+**" "**;**

**if(**CommonCB\_6**->**Checked **&&** CommonCB\_6**->**Enabled)

WhereClause **+=**"\nAND O.NAME LIKE '%"**+** CommonEB\_2**->**Text **+**"%' "**;**

**break;**

//------------FUNCTION IS CUT HERE!

void \_\_fastcall TRefFRM**::**QuerySaveBitBtnClick**(**TObject **\***Sender**)**

**{**

**if(**QuerySaveDialog**->**Execute**()**)

WriteReferenceToFile**(**QuerySaveDialog**->**FileName)**;**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TRefFRM**::**FormClose**(**TObject **\***Sender**,** TCloseAction **&**Action**)**

**{**

**if(**IBDataModule**->**ReferenceQRY**->**Active)

IBDataModule**->**ReferenceQRY**->**Active **=false;**

**}**

//---------------------------------------------------------------------------

**PersonFORM.h**

#ifndef PersonFORMH

#define PersonFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBCtrls.hpp>

#include <DBGrids.hpp>

#include <ExtCtrls.hpp>

#include <Grids.hpp>

//---------------------------------------------------------------------------

class TPersonsFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBGrid **\***IBPersonTblDBGrid**;**

TDBNavigator **\***PersonDBNavigator**;**

TGroupBox **\***PrinterTblGB**;**

TLabel **\***OrderLabelPerson**;**

TLabel **\***SearchLabelPerson**;**

TEdit **\***PersonTblSearchEB**;**

TComboBox **\***OrderComboPerson**;**

TComboBox **\***SearchComboPerson**;**

void \_\_fastcall OrderComboPersonSelect**(**TObject **\***Sender**);**

void \_\_fastcall PersonTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall IBPersonTblDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TPersonsFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TPersonsFRM **\***PersonsFRM**;**

//---------------------------------------------------------------------------

#endif

**PersonFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "PersonFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TPersonsFRM **\***PersonsFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TPersonsFRM**::**TPersonsFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TPersonsFRM**::**OrderComboPersonSelect**(**TObject **\***Sender**)**

**{**

**switch(**OrderComboPerson**->**ItemIndex)

**{**

**case**0**:** IBDataModule**->**IBPersonTbl**->**IndexName **=**"PersonTblLNameNDX"**;break;**

**case**1**:** IBDataModule**->**IBPersonTbl**->**IndexName **=**"PersonTblFNameNDX"**;break;**

**case**2**:** IBDataModule**->**IBPersonTbl**->**IndexName **=**"PersonTblSNameNDX"**;break;**

**case**3**:** IBDataModule**->**IBPersonTbl**->**IndexName **=**"PersonTblEgnNDX"**;break;**

**case**4**:** IBDataModule**->**IBPersonTbl**->**IndexName **=**"PersonTblPhoneNDX"**;break;**

**}**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TPersonsFRM**::**PersonTblSearchEBChange**(**TObject **\***Sender**)**

**{**

AnsiString FieldName **=**""**;**

TLocateOptions Opts**;**

Opts**.**Clear**();**

Opts **<<** loPartialKey**;**

Opts **<<** loCaseInsensitive**;**

**switch(**SearchComboPerson**->**ItemIndex**)**

**{**

**case**0**:** FieldName **=**"LNAME"**;break;**

**case**1**:** FieldName **=**"FNAME"**;break;**

**case**2**:** FieldName **=**"SNAME"**;break;**

**case**3**:** FieldName **=**"EGN"**;break;**

**case**4**:** FieldName **=**"PHONE"**;break;**

**}**

IBDataModule**->**IBPersonTbl**->**Locate**(**FieldName**,** PersonTblSearchEB**->**Text**,** Opts)**;**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TPersonsFRM**::**IBPersonTblDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBPersonTbl**->**State **==** dsInsert)

**||** (IBDataModule**->**IBPersonTbl**->**State **==** dsEdit))

IBDataModule**->**IBPersonTbl**->**Post**();**

**}**

//---------------------------------------------------------------------------

**ActRecFORM.h**

#ifndef ActRecFORMH

#define ActRecFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

#include <DBCtrls.hpp>

#include <ExtCtrls.hpp>

//---------------------------------------------------------------------------

class TActRecFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TGroupBox **\***CmptrTblGB**;**

TLabel **\***OrderLabelActivities**;**

TLabel **\***SearchLabelActivities**;**

TEdit **\***ActivitiesTblSearchEB**;**

TComboBox **\***OrderComboActivities**;**

TDBGrid **\***ActRecDBGrid**;**

TDBNavigator **\***ActRecDBNavigator**;**

void \_\_fastcall OrderComboActivitiesSelect**(**TObject **\***Sender**);**

void \_\_fastcall ActivitiesTblSearchEBChange**(**TObject **\***Sender**);**

void \_\_fastcall ActRecDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TActRecFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TActRecFRM **\***ActRecFRM**;**

//---------------------------------------------------------------------------

#endif

**ActRecFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "ActRecFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TActRecFRM **\***ActRecFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TActRecFRM**::**TActRecFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TActRecFRM**::**OrderComboActivitiesSelect**(**TObject **\***Sender**)**

**{**

**switch(**OrderComboActivities**->**ItemIndex)

**{**

**case**0**:** IBDataModule**->**IBActRecTbl**->**IndexName **=**""**;break;**

**case**1**:** IBDataModule**->**IBActRecTbl**->**IndexName **=**"ActRecTblNDX"**;break;**

**}**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TActRecFRM**::**ActivitiesTblSearchEBChange**(**TObject **\***Sender**)**

**{**

TLocateOptions Opts**;**

Opts**.**Clear**();**

Opts **<<** loPartialKey**;**

Opts **<<** loCaseInsensitive**;**

IBDataModule**->**IBActRecTbl**->**Locate**(**"DESCR"**,** ActivitiesTblSearchEB**->**Text**,** Opts)**;**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TActRecFRM**::**ActRecDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBActRecTbl**->**State **==** dsInsert**)**

**||** (IBDataModule**->**IBActRecTbl**->**State **==** dsEdit**)**)

IBDataModule**->**IBActRecTbl**->**Post**();**

**}**

//---------------------------------------------------------------------------

**ManufactFORM.h**

#ifndef ManufactFORMH

#define ManufactFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBCtrls.hpp>

#include <DBGrids.hpp>

#include <ExtCtrls.hpp>

#include <Grids.hpp>

#include <Buttons.hpp>

//---------------------------------------------------------------------------

class TManufactFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBNavigator **\***ManufactDBNavigator**;**

TDBGrid **\***ManufactDBGrid**;**

TGroupBox **\***ManufactGB**;**

TBitBtn **\***ManufacBitBtn**;**

void \_\_fastcall ManufacBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall ManufactDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TManufactFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TManufactFRM **\***ManufactFRM**;**

//---------------------------------------------------------------------------

#endif

**ManufactFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "ManufactFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TManufactFRM **\***ManufactFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TManufactFRM**::**TManufactFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TManufactFRM**::**ManufacBitBtnClick**(**TObject **\***Sender**)**

**{**

ManufactFRM**->**Close**();**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TManufactFRM**::**ManufactDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBManufactTbln**->**State **==** dsInsert)**||**

(IBDataModule**->**IBManufactTbln**->**State **==** dsEdit))

IBDataModule**->**IBManufactTbln**->**Post**();**

**}**

//---------------------------------------------------------------------------

**DepFORM.h**

#ifndef DepFORMH

#define DepFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBCtrls.hpp>

#include <DBGrids.hpp>

#include <ExtCtrls.hpp>

#include <Grids.hpp>

#include <Buttons.hpp>

//---------------------------------------------------------------------------

class TDepFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBNavigator **\***DepartDBNavigator**;**

TDBGrid **\***DepartDBGrid**;**

TGroupBox **\***DepartGB**;**

TBitBtn **\***DepartBitBtn**;**

void \_\_fastcall DepartBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall DepartDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TDepFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TDepFRM **\***DepFRM**;**

//---------------------------------------------------------------------------

#endif

**DepFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "DepFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TDepFRM **\***DepFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TDepFRM**::**TDepFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TDepFRM**::**DepartBitBtnClick**(**TObject **\***Sender**)**

**{**

DepFRM**->**Close**();**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TDepFRM**::**DepartDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBDepartTbln**->**State **==** dsInsert)**||**

(IBDataModule**->**IBDepartTbln**->**State **==** dsEdit))

IBDataModule**->**IBDepartTbln**->**Post**();**

**}**

//---------------------------------------------------------------------------

**PosFORM.h**

#ifndef PosFORMH

#define PosFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBCtrls.hpp>

#include <DBGrids.hpp>

#include <ExtCtrls.hpp>

#include <Grids.hpp>

#include <Buttons.hpp>

//---------------------------------------------------------------------------

class TPosFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBNavigator **\***PositionDBNavigator**;**

TDBGrid **\***PositionDBGrid**;**

TGroupBox **\***PositionGB**;**

TBitBtn **\***PositionBitBtn**;**

void \_\_fastcall PositionBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall PositionDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TPosFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TPosFRM **\***PosFRM**;**

//---------------------------------------------------------------------------

#endif

**PosFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "PosFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TPosFRM **\***PosFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TPosFRM**::**TPosFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TPosFRM**::**PositionBitBtnClick**(**TObject **\***Sender**)**

**{**

PosFRM**->**Close**();**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TPosFRM**::**PositionDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBPositionTbln**->**State **==** dsInsert)**||**

(IBDataModule**->**IBPositionTbln**->**State **==** dsEdit))

IBDataModule**->**IBPositionTbln**->**Post**();**

**}**

//---------------------------------------------------------------------------

**OfficeFORM.h**

#ifndef OfficeFORMH

#define OfficeFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <Buttons.hpp>

#include <DBCtrls.hpp>

#include <ExtCtrls.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

//---------------------------------------------------------------------------

class TOfficeFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBNavigator **\***OfficeDBNavigator**;**

TGroupBox **\***OfficeGB**;**

TBitBtn **\***OfficeBitBtn**;**

TDBGrid **\***OfficeDBGrid**;**

void \_\_fastcall OfficeBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall OfficeDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TOfficeFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TOfficeFRM **\***OfficeFRM**;**

//---------------------------------------------------------------------------

#endif

**OfficeFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "OfficeFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TOfficeFRM **\***OfficeFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TOfficeFRM**::**TOfficeFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TOfficeFRM**::**OfficeBitBtnClick**(**TObject **\***Sender**)**

**{**

OfficeFRM**->**Close**();**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TOfficeFRM**::**OfficeDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBOfficeTbl**->**State **==** dsEdit)**||**

(IBDataModule**->**IBOfficeTbl**->**State **==** dsInsert))

IBDataModule**->**IBOfficeTbl**->**Post**();**

**}**

//---------------------------------------------------------------------------

**ActTypeFORM.h**

#ifndef ActTypeFORMH

#define ActTypeFORMH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <DBCtrls.hpp>

#include <ExtCtrls.hpp>

#include <Buttons.hpp>

#include <DBGrids.hpp>

#include <Grids.hpp>

//---------------------------------------------------------------------------

class TActTypeFRM **:**public TForm

**{**

\_\_published**:**// IDE-managed Components

TDBNavigator **\***ActTypeDBNavigator**;**

TGroupBox **\***ActTypeGB**;**

TBitBtn **\***ActTypeBitBtn**;**

TDBGrid **\***ActTypeDBGrid**;**

void \_\_fastcall ActTypeBitBtnClick**(**TObject **\***Sender**);**

void \_\_fastcall ActTypeDBGridExit**(**TObject **\***Sender**);**

private**:**// User declarations

public**:**// User declarations

\_\_fastcall TActTypeFRM**(**TComponent**\*** Owner**);**

**};**

//---------------------------------------------------------------------------

extern PACKAGE TActTypeFRM **\***ActTypeFRM**;**

//---------------------------------------------------------------------------

#endif

**ActTypeFORM.cpp**

#include <vcl.h>

#pragma hdrstop

#include "ActTypeFORM.h"

#include "DModFRM.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TActTypeFRM **\***ActTypeFRM**;**

//---------------------------------------------------------------------------

\_\_fastcall TActTypeFRM**::**TActTypeFRM**(**TComponent**\*** Owner**)**

**:** TForm**(**Owner**)**

**{**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TActTypeFRM**::**ActTypeBitBtnClick**(**TObject **\***Sender**)**

**{**

ActTypeFRM**->**Close**();**

**}**

//---------------------------------------------------------------------------

void \_\_fastcall TActTypeFRM**::**ActTypeDBGridExit**(**TObject **\***Sender**)**

**{**

**if((**IBDataModule**->**IBActTypeTbln**->**State **==** dsInsert)**||**

(IBDataModule**->**IBActTypeTbln**->**State **==** dsEdit))

IBDataModule**->**IBActTypeTbln**->**Post**();**

**}**

//---------------------------------------------------------------------------