

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

МАИ

Институт № 8 «Информационные технологии и прикладная математика»

Кафедра 805 «Математическая кибернетика»

Курсовая Работа

по дисциплине

БАЗЫ ДАННЫХ

Работу выполнили

студенты группы М8О-303Б-18 и М8О-304Б-18

Хахин М.С., Мукин Ю.Д. и Сыроежкин К.Г.

Работу приняли

кандидат технических наук,

Доцент Киндинова В. В., Кузнецова Е.В.

МОСКВА, 2020

1. Задание

1.1 Анализ предметной области

С ростом рынка компьютерных комплектующих стало сложнее вести складской учет на предприятиях занимающихся сборкой ПК. Это вылилось в расхождение ожидаемых расходов с реальными расходами, в проблемы с учетом гарантийных случаев и в воровство на предприятии.

Накопившийся объем возможных конфигураций сильно усложняет задачу подбора комплектующих для покупателей. Раньше, в связи с ограниченностью развития технологий, полный список доступных комплектующих легко умещался в небольшой журналчик, и не составляло труда подобрать конфигурацию. Так как с каждым годом технологии шагают вперед, объем журналов вырос до неприемлемых размеров, что привело к их почти полному исчезновению. Поэтому появилась необходимость в создании удобного инструмента для проектирования конфигурации вычислительных систем. Таким инструментом стала реляционная база данных.

База данных имеет множество преимуществ перед бумажными носителями. Основные преимущества БД:

- Удобное иерархическое устройство;
- Возможность проверки комплектующих на совместимость;
- Моментальный доступ к информации;
- Возможность получить необходимую информацию в любой момент из любой точки мира по сети;

Наше приложение позволяет работать с базой данных трем типам пользователей - клиент, сборщик, менеджер. У каждого типа свои потребности в работе с БД. Так, к примеру, клиент должен знать, какие готовые конфигурации предприятие может предложить, а также полный список доступных комплектующих и состояние своего заказа. Сборщик в свою очередь должен иметь доступ к списку доступных для выполнения заказов и их конфигурациям. Менеджер имеет доступ ко всем существующим таблицам, что позволяет эффективнее управлять предприятием.

1.2 Постановка задачи

Разработать прикладное программное обеспечение для компании по осуществлению сборки компьютеров, объединяющее в себе 3 приложения

для каждой из категорий пользователей: клиент, сборщик, менеджер. Оно должно иметь следующий функционал:

Приложение для клиента:

- Внести свою конфигурацию компьютера в базу данных.
- Посмотреть существующую конфигурацию, которая удовлетворяет требованиям клиента.
- Посмотреть доступные запчасти.
- Информацию по сборке компьютера по его конфигурации.

Приложение для сборщика:

- Посмотреть существующую конфигурацию для последующей сборки.
- Занести собранный компьютер в базу данных.

Приложение для менеджера:

- Информацию о собранных компьютерах.
- Информацию о запчастях на складах
- Информацию о сборщиках.

Таблицы и схема данных

В разработанной базе данных присутствуют таблицы: «case», «disc», «mother», «power_supply», «ram», «gpu», «cpu», «components», «final_product», «assemblers».

В каждом финальном продукте (final_product) содержится pc_id (номер компьютера), ass_id (номер сборщика), conf_id (номер конфигурации), ok (исправность), ass_date (дата сборки).

При сборке компьютера, покупатель может выбрать:

1) case (корпус): case_id (номер корпуса), standart (форм-фактор корпуса), coast (цена);

2) disk (жесткий диск): disk_id (номер диска), size (объем в Гб), type (ssd/hdd), coast (цена);

3) motherboard (материнская плата): motherboard_id (номер материнской платы), soket (тип разъема), chipset (код южного моста), coast (цена);

4) power_supply (блок питания): power_supply_id (номер блока питания), power (мощность в Вт), standart (форм-фактор), coast (цена);

5) ram (оперативная память): ram_id (номер оперативной памяти), quantity (частота), size (объём в Гб), coast (цена);

6) gpu (видео карта): gpu_id (номер видео карты), cores (кол-во ядер), gt (поддерживается ли трассировка лучей), frequency (частота ядер), coast (цена);

7) cpu (процессор): cpu_id (номер процессора), socket (код разъема на материнской плате), frequency (частота ядер), cores (кол-во ядер), threads (кол-во потоков), coast (цена).

Все компоненты хранятся в таблице components: conf_id (номер конфигурации), type (тип по назначению), case_id (номер корпуса), disk_id (номер диска), motherboard_id (номер материнской платы), power_supply_id (номер блока питания), ram_id (номер оперативной памяти), gpu_id (номер видео карты), cpu_id (номер процессора).

У каждого сборщика (assemblers) есть: ass_id (номер сборщика), name (имя), pc_count (кол-во собранных компьютеров), marriag_rate (рейтинг сборщика).

Поля таблиц и их характеристики представлены в таблице 1.

Таблица 1. Поля таблиц базы данных и их характеристики

Поле	Тип данных	Длина	Дополнительно
«case»			
case_id	Intefer	4	
standart	Character	10	
coast	Double	8	
«disk»			
disk_id	Integer	4	

size	Numeric	10	
type	Character	3	
coast	Double	8	
«motherboard»			
motherboard_id	Iteger	4	
soket	Character	10	
chipzet	Character	10	
coast	Double	8	
«power_supply»			
power_suply_id	Integer	4	
power	Integer	4	
standart	Character	10	
coast	Double	8	
«ram»			
ram_id	Integer	4	
quanntity	Numeric	2	
size	Numeric	10	
coast	Character	10	
«gpu»			
gpu_id	Integer	4	
cores	Memo	4	
rt	Character	1	

frequency	Numeric	10	
coast	Double	8	
«cpu»			
cpu_id	Integer	4	
socket	Integer	4	
frequency	Numeric	4	
cores	Numeric	2	
threads	Numeric	3	
coast	Double	8	
«components»			
conf_id	Integer	4	
type	Character	10	
cpu_id	Character	10	
gpu_id	Character	10	
ram_id	Character	10	
disk_id	Character	10	
motherboard_id	Character	10	
power_supply_id	Character	10	
case_id	Character	10	
«final_product»			
pc_id	Integer	4	
ass_id	Integer	4	
conf_id	Integer	4	

ok	Character	1	
ass_date	Character	1	
«assemblers»			
ass_id	Integer	4	
name	Character	10	
pc_count	Integer	4	
marriag_rate	Integer	4	

2. Проект системы

2.1 Информационная часть:

Схема в ERWin

Результаты построения разработанной базы данных в ERWin

продемонстрированы на рис. 1 – 3.

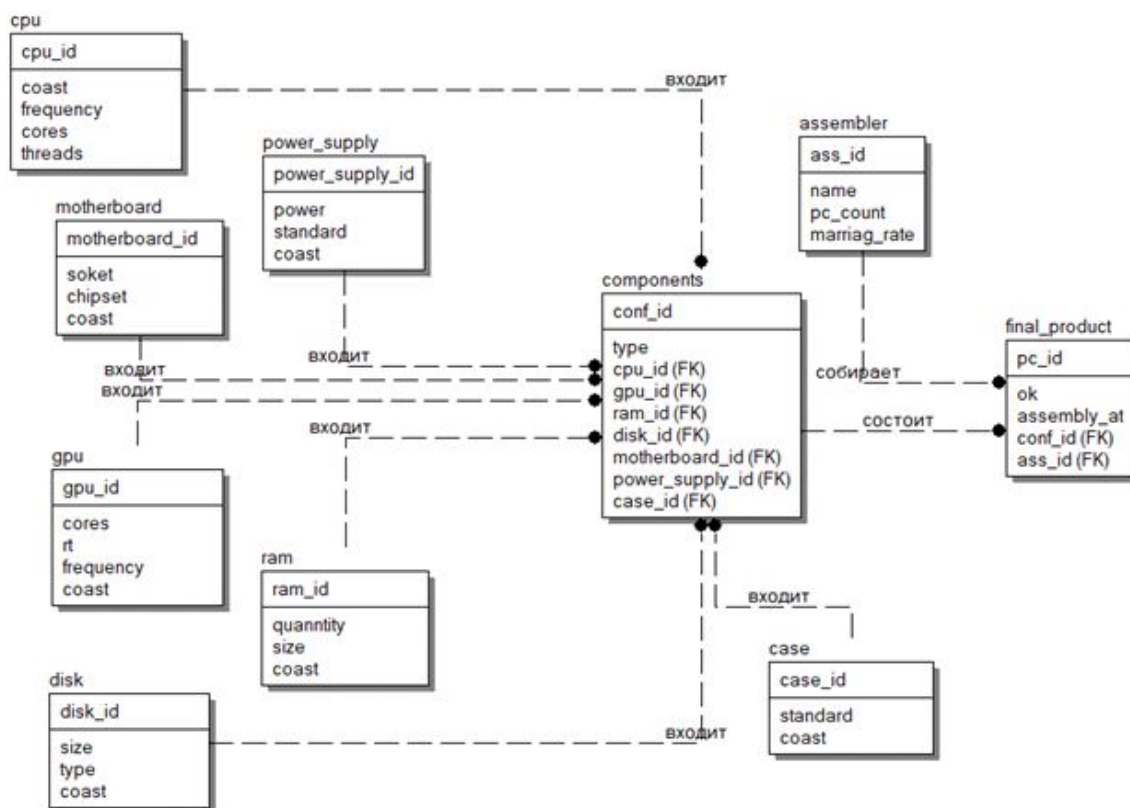


Рис. 1. Логическая схема в ERWin

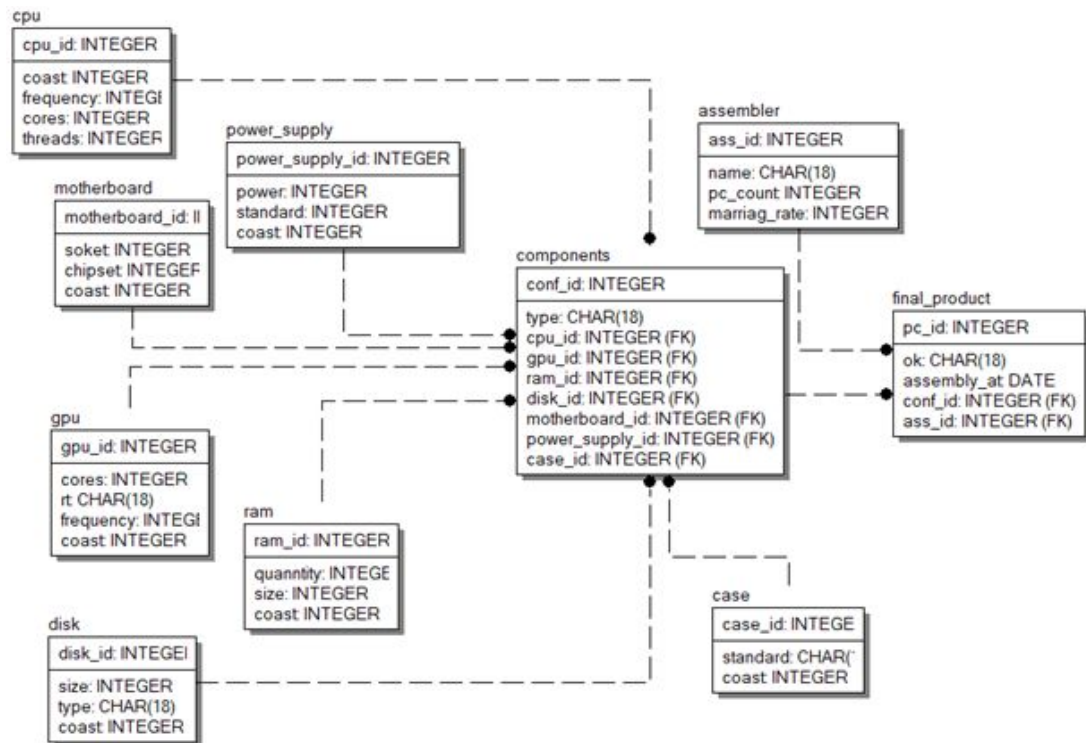


Рис. 2. Физическая схема в ERWin

```

CREATE TABLE assembler
(
    ass_id          CHAR(18) NOT
NULL ,
    name           CHAR(18) NULL ,
    pc_count       CHAR(18) NULL
,
    marriag_rate   CHAR(18)
NULL
);
  
```

```

CREATE UNIQUE INDEX
XPKassembler ON assembler
(ass_id ASC);
ALTER TABLE assembler
ADD CONSTRAINT
XPKassembler PRIMARY KEY (ass_id);
  
```

```

CREATE TABLE case
(
    case_id        CHAR(18) NOT
NULL ,
    standard       CHAR(18) NULL
,
    coast          CHAR(18) NULL
);
  
```

```

CREATE UNIQUE INDEX XPKcase ON
case
(case_id ASC);
  
```

```

ALTER TABLE case
ADD CONSTRAINT XPKcase
PRIMARY KEY (case_id);
  
```

```

CREATE TABLE components
(
    conf_id          CHAR(18) NOT
NULL ,
    type            CHAR(18) NULL ,
    cpu_id           INTEGER NULL ,
    gpu_id           CHAR(18) NOT
NULL ,
    ram_id           CHAR(18) NULL
,
    disk_id          CHAR(18) NULL ,
    motherboard_id   CHAR(18)
NOT NULL ,
    power_supply_id  CHAR(18)
NOT NULL ,
    case_id          CHAR(18) NOT
NULL
);

```

```

CREATE UNIQUE INDEX
XPKcomponents ON components
(conf_id ASC);

```

```

ALTER TABLE components
    ADD CONSTRAINT
XPKcomponents PRIMARY KEY
(conf_id);

```

```

CREATE TABLE cpu
(
    cpu_id           INTEGER NOT
NULL ,
    frequency        INTEGER NULL
,
    cores            INTEGER NULL ,
    threads          CHAR(18) NULL ,
    coast            INTEGER NULL
);

```

```

CREATE UNIQUE INDEX XPKcpu ON
cpu
(cpu_id ASC);

```

```

ALTER TABLE cpu
    ADD CONSTRAINT XPKcpu
PRIMARY KEY (cpu_id);

```

```

CREATE TABLE disk
(
    disk_id          CHAR(18) NOT
NULL ,
    size             CHAR(18) NULL ,
    type             CHAR(18) NULL ,
    coast            CHAR(18) NULL
);

```

```

CREATE UNIQUE INDEX XPKdisk ON
disk
(disk_id ASC);

```

```

ALTER TABLE disk
    ADD CONSTRAINT XPKdisk
PRIMARY KEY (disk_id);

```

```

CREATE TABLE final_product
(
    pc_id            CHAR(18) NOT
NULL ,
    ok               CHAR(18) NULL ,
    assembly_at      DATE NULL ,
    conf_id          CHAR(18) NULL
,
    ass_id           CHAR(18) NOT
NULL
);

```

```
CREATE UNIQUE INDEX
XPKfinal_product ON final_product
(pc_id ASC);
```

```
ALTER TABLE final_product
ADD CONSTRAINT
XPKfinal_product PRIMARY KEY
(pc_id);
```

```
CREATE TABLE gpu
(
    gpu_id          CHAR(18) NOT
NULL ,
    cores           CHAR(18) NULL ,
    rt              CHAR(18) NULL ,
    frequency       CHAR(18)
NULL ,
    coast           CHAR(18) NULL
);
```

```
CREATE UNIQUE INDEX XPKgpu ON
gpu
(gpu_id ASC);
```

```
ALTER TABLE gpu
ADD CONSTRAINT XPKgpu
PRIMARY KEY (gpu_id);
```

```
CREATE TABLE motherboard
(
    motherboard_id  CHAR(18)
NOT NULL ,
    soket           CHAR(18) NULL ,
    chipset         CHAR(18) NULL ,
    coast           CHAR(18) NULL
);
```

```
CREATE UNIQUE INDEX
XPKmotherboard ON motherboard
(motherboard_id ASC);
```

```
ALTER TABLE motherboard
ADD CONSTRAINT
XPKmotherboard PRIMARY KEY
(motherboard_id);
```

```
CREATE TABLE power_supply
(
    power_supply_id CHAR(18)
NOT NULL ,
    power           CHAR(18) NULL
,
    standard        CHAR(18) NULL
,
    coast           CHAR(18) NULL
);
```

```
CREATE UNIQUE INDEX
XPKpower_supply ON power_supply
(power_supply_id ASC);
```

```
ALTER TABLE power_supply
ADD CONSTRAINT
XPKpower_supply PRIMARY KEY
(power_supply_id);
```

```
CREATE TABLE ram
(
    ram_id          CHAR(18) NOT
NULL ,
    quanntity       CHAR(18) NULL
,
    size            CHAR(18) NULL ,
    coast           CHAR(18) NULL
)
```

Рис. 3. Описание сгенерированной базы данных на SQL

Схема в FoxPro

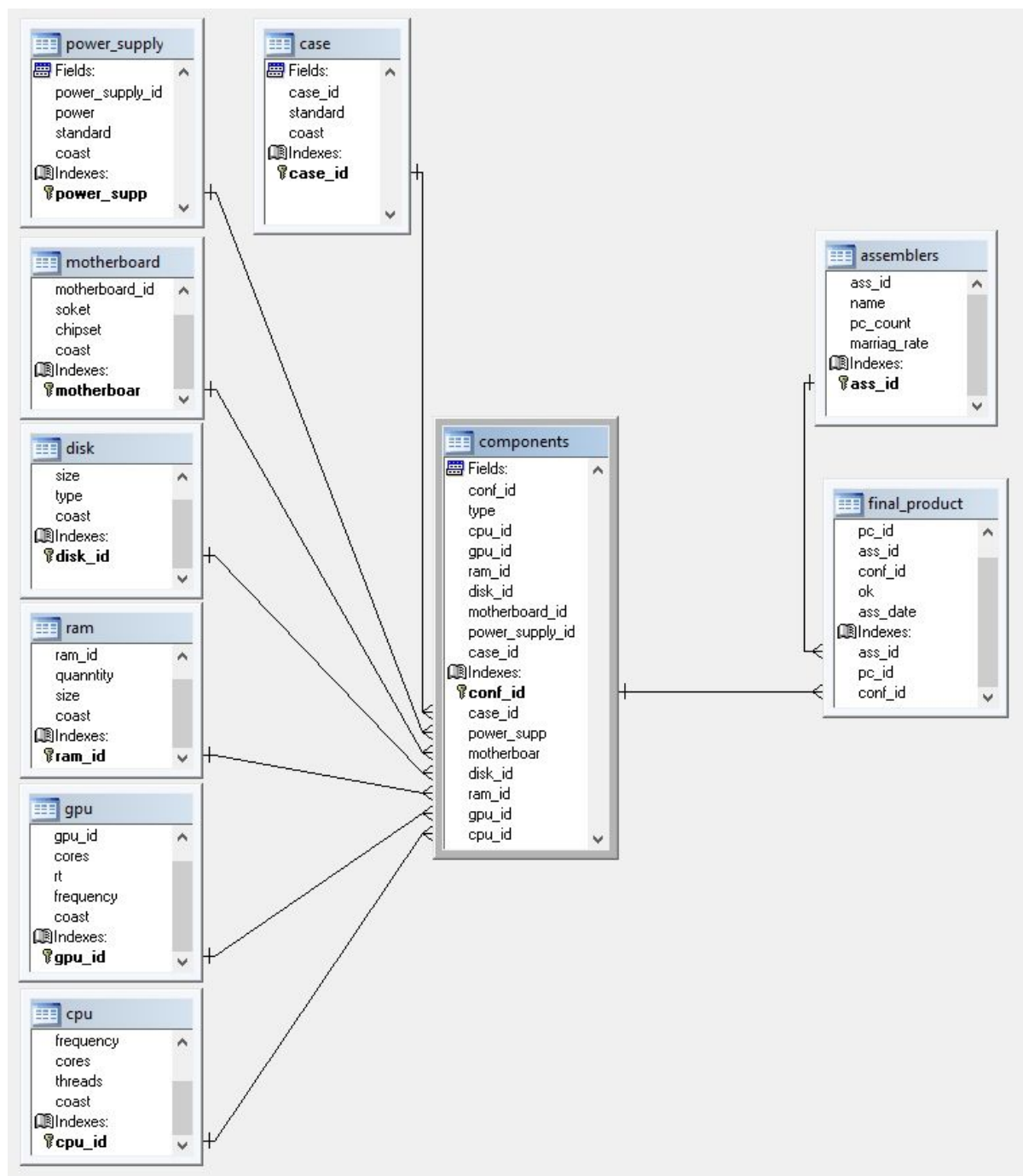
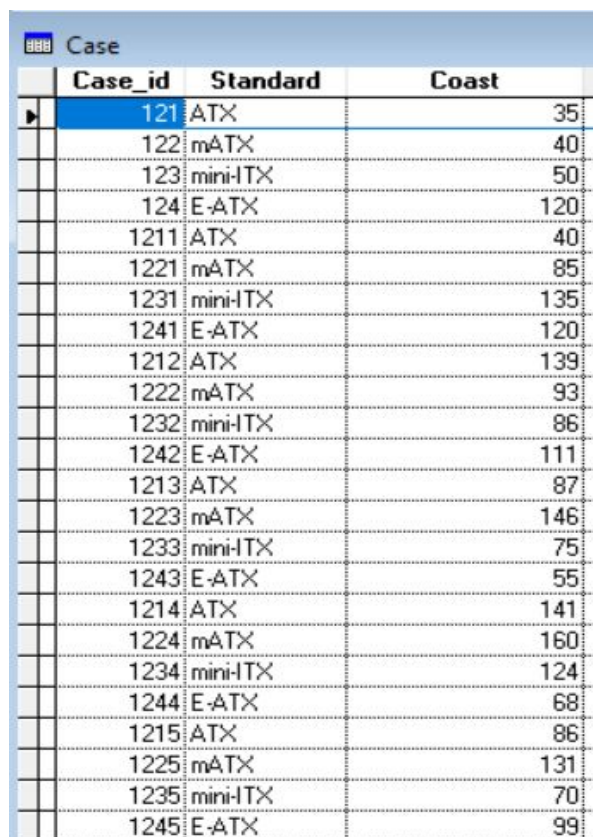


Рис. 4. Схема базы данных в FoxPro

3. Реализация проекта

3.1 Заполненная БД

Примеры заполнения таблиц приведены на рис. 5– 14.



Case_id	Standard	Coast
121	ATX	35
122	mATX	40
123	mini-ITX	50
124	E-ATX	120
1211	ATX	40
1221	mATX	85
1231	mini-ITX	135
1241	E-ATX	120
1212	ATX	139
1222	mATX	93
1232	mini-ITX	86
1242	E-ATX	111
1213	ATX	87
1223	mATX	146
1233	mini-ITX	75
1243	E-ATX	55
1214	ATX	141
1224	mATX	160
1234	mini-ITX	124
1244	E-ATX	68
1215	ATX	86
1225	mATX	131
1235	mini-ITX	70
1245	E-ATX	99

Рис. 5. Таблица «case»

Disk				
	Disk_id	Size	Type	Coast
▶	115121	512	SSD	120
	112561	256	SSD	60
	1110242	1024	HHD	70
	1110241	1024	SSD	240
	1120481	2048	SSD	480
	111281	128	SSD	30
	1140961	4096	SSD	960
	1115361	1536	SSD	360
	1125681	2568	SSD	600
	1146081	4608	SSD	1080
	1181921	8192	SSD	1920
	115122	512	HHD	35
	112562	256	HHD	18
	1120482	2048	HHD	140
	111282	128	HHD	9
	1140962	4096	HHD	280
	1115362	1536	HHD	105
	1125682	2568	HHD	175
	1146082	4608	HHD	315
	1181922	8192	HHD	560

Рис. 6. Таблица «disk»

Motherboard				
	Motherboard_id	Soket	Chipset	Coast
▶	14133665	1336	65	70
	14120085	1200	85	120
	14105697	1056	97	200
	14133667	1336	67	75
	14133631	1336	31	35
	141200270	1200	270	330
	141200250	1200	250	300
	141056100	1056	100	150
	14105625	1056	25	30
	141151150	1151	150	170
	141151170	1151	170	180
	141151110	1151	110	165

Рис. 7. Таблица «motherboard»

Power_supply				
	Power_supply_id	Power	Standard	Coast
	105501	550	ATX	80
	106501	650	ATX	100
	104502	450	SFX	85
	104501	450	ATX	60
	107501	750	ATX	120
	108501	850	ATX	140
	109501	950	ATX	160
	1010501	1050	ATX	180
	1011501	1150	ATX	200
	1012501	1250	ATX	220
	1013501	1350	ATX	240
	1014501	1450	ATX	260
	1015501	1550	ATX	280
	104002	400	SFX	75
	105002	500	SFX	95
	105502	550	SFX	105
	106002	600	SFX	115
	106502	650	SFX	125
	107002	700	SFX	135
	107502	750	SFX	145

Рис. 8. Таблица «power_supply»

Ram				
	Ram_id	Quantity	Size	Coast
	1336008	36	8	105
	13400016	40	16	125
	1324008	24	8	45
	13200016	20	16	25
	1320008	20	8	20
	1322008	22	8	30
	13220016	22	16	35
	13240016	24	16	40
	1326008	26	8	50
	13260016	26	16	55
	1328008	28	8	60
	13280016	28	16	65
	1330008	30	8	70
	13300016	30	16	75
	1332008	32	8	80
	13320016	32	16	85
	1334008	34	8	90
	13340016	34	16	95
	13360016	36	16	100
	1338008	38	8	110
	13380016	38	16	115
	1340008	40	8	120

Рис. 9. Таблица «ram»

Gpu					
	Gpu_id	Cores	Rt	Frequency	Coast
▶	159801	3564	n	1000	399
	1520702	2578	y	2500	450
	1510301	1567	n	1500	230
	159701	2564	n	1000	350
	159601	1996	n	1000	300
	1520602	2078	y	2000	320
	1516601	2078	n	2000	299
	1516501	2000	n	2000	250
	1520802	3000	y	2000	550
	1530702	2500	y	1900	600
	1530602	2128	y	1900	500
	1530802	3500	y	1900	700

Рис. 10. Таблица «gpu»

Cpu						
	Cpu_id	Socket	Frequency	Cores	Threads	Coast
▶	1613368	1336	4700	8	8	250
	1612008	1200	3400	8	16	300
	1610564	1056	4300	4	8	200
	1611514	1151	3500	4	8	150
	1611518	1151	3700	8	16	200
	16115112	1151	4000	12	24	250
	16115116	1151	4200	16	32	300
	1610568	1056	4600	8	8	250
	16105612	1056	4700	12	12	300
	16105616	1056	4900	16	16	350
	1612004	1200	3100	4	8	250
	16120012	1200	3600	12	24	350
	16120016	1200	3900	16	32	400
	1613364	1336	4300	4	4	200
	16133612	1336	5000	12	12	300
	16133616	1336	5300	16	16	350

Рис. 11. Таблица «cpu»

Components									
	Conf_id	Type	Cpu_id	Gpu_id	Ram_id	Disk_id	Motherboard_id	Power_supply_id	Case_id
▶	1	gaming	1613368	159801	1336008	115121	14133665	105501	121
	2	wkstation	1612008	1510301	13240032	115121	14120085	106501	121
	3	multimedia	1610564	159801	1336008	1125621	14105697	105501	121
	4	gaming	16120016	1530802	1340001	1181921	14105697	1015501	1224
	5	wkstation	16115116	159601	13400016	118192	14115117	107501	1215
	6	multimedia	1610564	1516601	13200016	1110242	14105625	105501	1211
	7	gaming	1612001	1530702	13380016	1146081	141200270	1014501	1223
	8	wkstation	16120016	159701	13380016	1140961	141200270	10650	1245
	9	multimedia	1613364	1516501	1320008	115122	1413363	104501	1211

Рис. 12. Таблица «components»

Final_product					
Pc_id	Ass_id	Conf_id	Ok	Ass_date	
1	2	2	y	10/10/20	
2	1	3	y	10/12/20	
3	7	1	n	10/15/20	
4	4	6	y	10/18/20	
5	6	5	y	10/20/20	
6	6	7	y	10/23/20	
7	6	4	y	10/25/20	
8	3	9	y	10/28/20	
9	2	8	y	10/30/20	

Рис. 13. Таблица «final_product»

Assemblers				
Ass_id	Name	Pc_count	Marriag_rate	
1	Василий	23	7	
2	Сооронбой	34	10	
3	Шавкат	45	8	
4	Виталий	36	6	
5	Дмитрий	70	5	
6	Юрий	146	10	
7	Никита	47	1	

Рис. 14. Таблица «assemblers»

3.2.1 Отлаженная программа (формы)

При запуске программы нам представляется меню выбора “роли”. 3 роли : сборщик assembler, покупатель customer и менеджер manager.

Кнопка start! содержит код запускающий форму соответствующую выбранному типу пользователя.

КОД КНОПКИ:

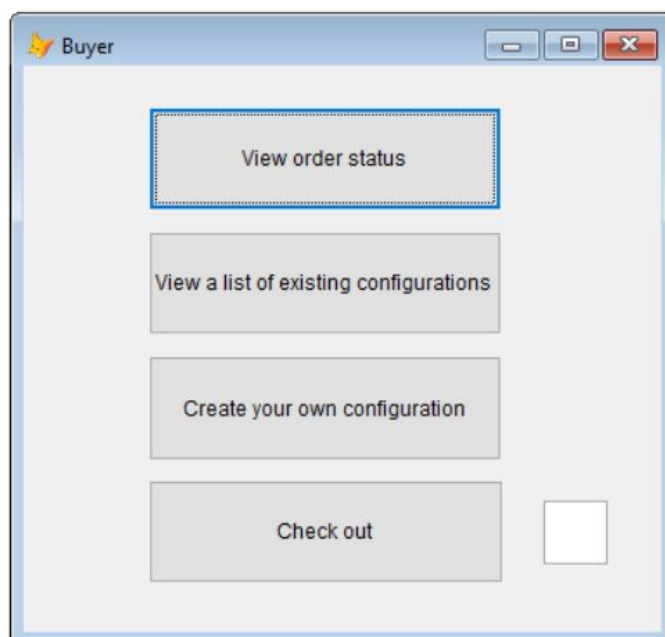
```
IF thisform.optiongroup1.Option1.Value = 1
    DO FORM forms/customer
ELSE
    IF thisform.optiongroup1.Option2.Value = 1
        DO FORM forms/manager
```

```
ELSE
    DO FORM forms/ass
ENDIF
ENDIF
thisform.release()
```

Далее рассмотрим в отдельности функции предоставляемые различным типам пользователей.

Покупатель.

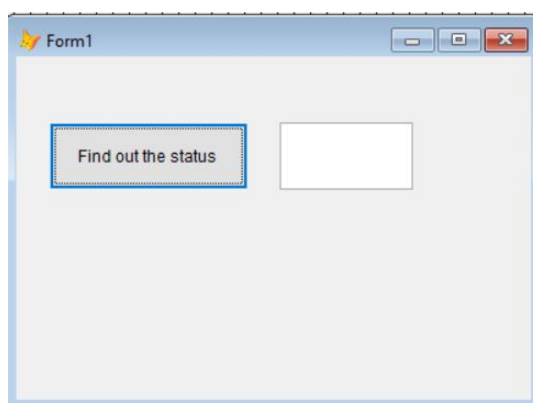
Рассмотрим сущность “Покупатель”. Когда мы заходим в сущность “Покупатель”, у нас есть 4 кнопки для дальнейших действий:



1) **“View order status”**- посмотреть статус заказа. Данная кнопка переводит нас в следующее окно, в котором мы вводим номер компьютера.

Код кнопки:

```
DO FORM Forms/Status;
```



Данная форма просит на вход id компьютера (pc_id) и в зависимости от готовности компьютера выводит соответствующий ответ.

Код кнопки:

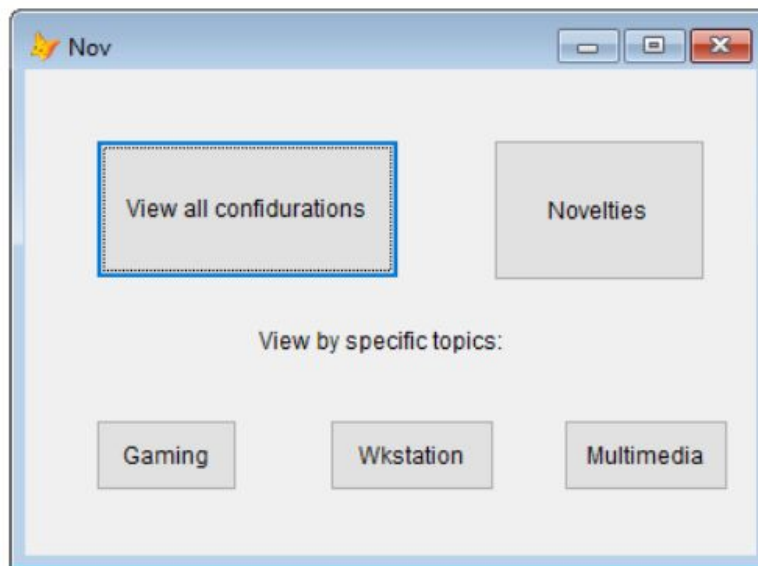
```
tx1 = thisform.text1.Text  
SELECT final_product.ok as ok;  
From final_product ;  
where final_product.pc_id = CAST(tx1 as I) into cursor tmp  
if tmp.ok = 'y'
```

```
MessageBox("Your order is verified and ready for delivery")
Else
MessageBox("Your order is not ready")
ENDIF
```

2) "View a list of existing configurations" - Посмотреть список существующих конфигураций. Данная кнопка переводит нас в следующее окно, в котором мы можем выбрать одну из нескольких видов поиска конфигураций.

Код кнопки:

```
DO FORM Forms/configuration
```



Данная форма дает покупателю выбрать разные виды списков с конфигурациями:

а) "View all configurations" - посмотреть список всех конфигураций.

Код кнопки:

```
SELECT components.*;
From components;
```

б) "Novelties" - конфигурации, которые отсортированы по дате сборки.

Код кнопки:

```
SELECT components.*, final_product.ass_date;
From components, final_product;
where components.conf_id = final_product.ass_id into cursor tmp
select tmp.*;
from tmp;
order by tmp.ass_date desc
```

в) "Gaming" - конфигурации, у которых тип "игровой".

Код кнопки:

```
SELECT components.*;
From components;
```

```
where components.type = 'gaming'
```

г) **“Wkstation”** - конфигурации, у которых тип “рабочая станция”.

Код кнопки:

```
SELECT components.*;  
From components;  
where components.type = 'wkstation'
```

д) **“Multimedia”** - конфигурации, у которых тип “мультимедийный”.

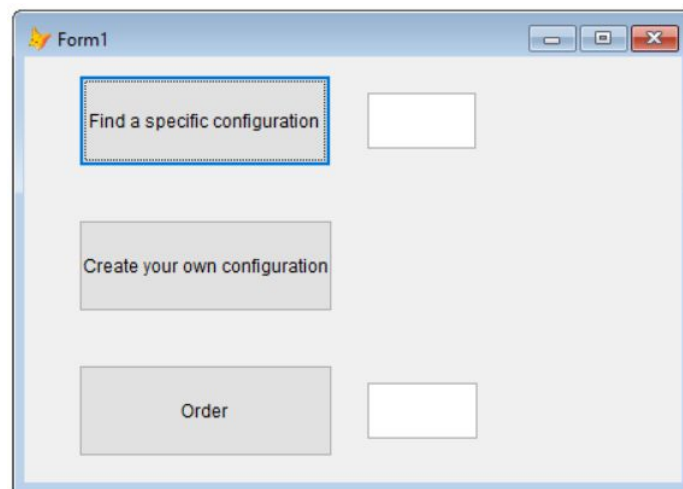
Код кнопки:

```
SELECT components.*;  
From components;  
where components.type = 'multimedia'
```

3) **”Create your own configuration”**- Создать собственную конфигурацию. Данная кнопка переводит нас в следующее окно, в котором мы можем выбрать одну из 3 функций данного окна.

Код кнопки:

```
DO FORM forms/create
```



Данная форма выполняет 3 функции:

а) **”Find a specific configuration”** - Найти выбранную конфигурацию по её номеру.

Код кнопки:

```
tx1 = thisform.text1.Text  
SELECT components.*;  
From components;  
where components.conf_id = CAST(tx1 as I)
```

б) **“Create your own configuration”** - создать свою собственную конфигурацию.

Данная кнопка переводит нас в следующую форму, которая нам дает возможность создать нашу собственную конфигурацию.

Код кнопки:

В данной форме мы выбираем компоненты из базы данных. Для удобства можно посмотреть каталог каждого комплектующего. Для добавления конфигурации в базу данных нужно нажать на кнопку “Add your configuration”

Код для этой кнопки: GO bottom

```
tx2 = thisform.combo2.value
tx3 = thisform.combo3.value
tx4 = thisform.combo4.value
tx5 = thisform.combo5.value
tx6 = thisform.combo6.value
tx7 = thisform.combo7.value
tx8 = thisform.combo8.value
tx9 = thisform.combo9.value
select components.conf_id;
from components;
where components.conf_id = (select MAX(components.conf_id) from components) into cursor tx1
INSERT INTO components
(conf_id,type,cpu_id,gpu_id,ram_id,disk_id,motherboard_id,power_supply_id,case_id);
VALUES (tx1.conf_id + 1, tx2 , cast(tx3 as I), cast(tx4 as I), cast(tx5 as I),cast(tx6 as I),cast(tx7 as I),cast(tx8 as I),cast(tx9 as I))
select components.conf_id as cunt;
from components;
where components.conf_id = tx1.conf_id + 1 into cursor cf
nm = STR(cf.cunt)
if cf.cunt = tx1.conf_id + 1
MESSAGEBOX("Sucessful. Your conf_id is: "+nm)
else
MESSAGEBOX("Fail")
ENDIF
```

Код для просмотра одного из каталогов (для остальных аналогично):

```
SELECT cpu.*;  
From cpu;
```

в) “Order” - заказать. Для заказа мы вводим id выбранной конфигурации и получаем в ответ pc_id, по которому в дальнейшем мы можем отслеживать наш заказ.

Код кнопки:

```
tx1 = thisform.text2.Text  
select final_product.pc_id;  
from final_product;  
where final_product.pc_id = (select MAX(final_product.pc_id) from final_product) into cursor tx2  
insert into final_product(pc_id,conf_id);  
values (tx2.pc_id+1,cast(tx1 as I))  
select final_product.pc_id as pc;  
from final_product;  
where final_product.pc_id = tx2.pc_id + 1 into cursor cf  
nm = STR(cf.pc)  
if cf.pc = tx2.pc_id + 1  
MESSAGEBOX("Sucessful. Your pc_id is: "+nm)  
else  
MESSAGEBOX("Fail")  
ENDIF
```

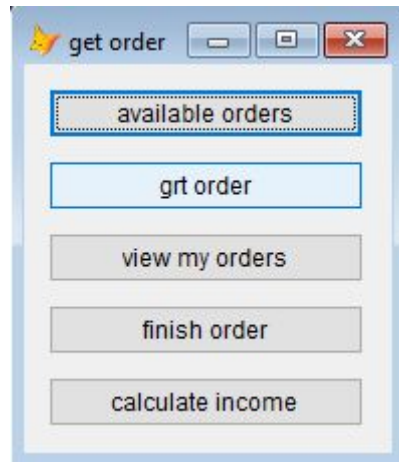
4) ”Check out” - Выдать чек покупателю. Данная кнопка выдает чек покупателю в зависимости от введенного id компьютера.

Код кнопки:

```
tx1 = thisform.text1.Text  
SELECT fp.pc_id as pc_id,fp.conf_id,fp.ass_date as ass_date , (gpu.coast + cpu.coast + ram.coast +  
dk.coast + ce.coast + ps.coast + mb.coast) as coast, components.case_id as case_id, components.ram_id  
as ram_id, components.power_supply_id as power_supply_id,components.motherboard_id as  
motherboard_id ,components.gpu_id as gpu_id,components.disk_id as disk_id,components.cpu_id as  
cpu_id ,gpu.coast as gpu_coast , cpu.coast as cpu_coast , ram.coast as ram_coast, dk.coast as dk_coast ,  
ce.coast as ce_coast , ps.coast as ps_coast , mb.coast as mb_coast;  
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,  
motherboard as mb;  
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND  
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND  
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id into cursor  
tmp  
SELECT tmp.*;  
from tmp;  
WHERE tmp.pc_id = CAST(tx1 AS I) into cursor tmp1  
SELECT tmp1  
REPORT FORM REPORTS/cheque preview
```

Сборщик.

При входе в систему как “сборщик” запустится форма содержащая 5 кнопок:



- 1) **“available orders”** - позволяет ознакомиться с списком сформированных заказов, которые еще никто не выполняет.

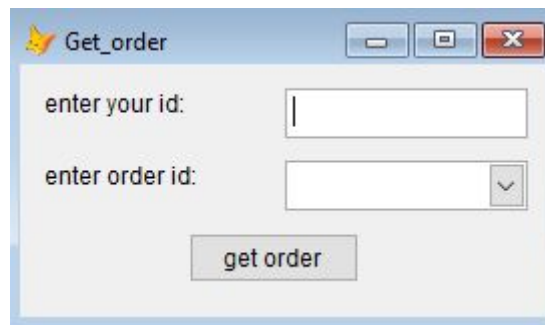
Код кнопки:

```
SELECT final_product.*, cm.type, cm.cpu_id, cm.gpu_id, cm.ram_id, cm.disk_id, cm.motherboard_id,
cm.power_supply_id, cm.case_id FROM final_product, components as cm;
WHERE final_product.ass_id = 0 AND final_product.conf_id = cm.conf_id
```

- 2) **“get order”** - вызывает вспомогательную форму, с помощью которой сборщик может закрепить заказ за собой.

Код кнопки:

DO FORM Forms/getting_order



В данной форме пользователь может ввести свой id в соответствующее поле и выбрать заинтересовавший его заказ из списка доступных, а затем обновить его статус в таблице нажав кнопку “get order”.

Код кнопки:

```
GO bottom
tx1 = thisform.text1.Text
tx2 = thisform.Combo1.Value
UPDATE final_product;
SET final_product.ass_id = CAST(tx1 AS I);
WHERE final_product.pc_id = CAST(tx2 AS I) AND final_product.ass_id = 0
SELECT final_product.ass_id as ai;
FROM final_product;
```



```

WHERE final_product.pc_id = CAST(tx2 AS I) INTO CURSOR tmp
IF tmp.ai = CAST(tx1 AS I)
    MESSAGEBOX("Sucessful")
ELSE
    MESSAGEBOX("FAIL")
ENDIF
SELECT final_product.pc_id;
from final_product;
WHERE final_product.ass_id = 0 INTO CURSOR tmp
thisform.combo1.RowSource = "tmp.pc_id"

```

3) **“view my orders”** - вызывает форму предназначенную для мониторинга статуса принятых сборщиком заказов.

Код кнопки:

DO FORM Forms/viewmyorders

В этой форме, введя свой id в соответствующее поле, по нажатию кнопки “view” сборщик может просмотреть принятые заказы. С помощью переключателя “all/only not finished” осуществляется фильтрация выдачи.

Код кнопки:

```

GO bottom
tx1 = thisform.text1.Text
SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast +
mb.coast) as coast;
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,
motherboard as mb;
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id into cursor
tmp

```

```

Thisform.grid1.columncount=-1
Thisform.grid1.recordsourcetype=1
IF thisform.optiongroup1.Option1.Value = 1
    SELECT final_product.*, tmp.coast;
    FROM tmp, final_product;
    WHERE tmp.pc_id = final_product.pc_id AND final_product.ass_id = CAST(tx1 AS I) into
cursor tmp2
ELSE
    SELECT final_product.*, tmp.coast;
    FROM tmp, final_product;
    WHERE tmp.pc_id = final_product.pc_id AND final_product.ass_id = CAST(tx1 AS I) AND
final_product.ok = '' into cursor tmp2
ENDIF
Thisform.grid1.recordsource = 'tmp2'

```

4) **“finish order”** - запускает форму позволяющую сборщику завершить выбранный заказ.

Код кнопки:

DO FORM Forms/finish_order

	Pc_id	Ass_id	Conf_id	Ok	Ass_date
	1	2	2	y	10/10/20
	2	1	3	y	10/12/20
	3	7	1	n	10/15/20
	4	4	6	y	10/18/20
	5	6	5	y	10/20/20
	6	6	7	y	10/23/20
	7	6	4	v	10/25/20

Для изменения статуса заказа сборщику необходимо ввести свой id, id заказа, дату сдачи заказа в соответствующие поля, а так же отметить его состояние на момент сдачи с помощью переключателя, после чего по нажатию кнопки “update status” данные в таблице будут обновлены. Отслеживать статус заказа можно в таблице размещенной на форме.

Код кнопки:

```

tx1 = thisform.text1.Text
tx2 = thisform.text2.Text
tx3 = thisform.text3.Text
Thisform.grid1.columncount=-1
Thisform.grid1.recordsourcetype=1
IF thisform.optiongroup1.Option1.Value = 1
    UPDATE final_product;
    SET final_product.ass_date = CAST(tx3 AS D), final_product.ok = 'y';
    WHERE final_product.pc_id = CAST(tx2 AS I) AND final_product.ass_id = CAST(tx1 AS I)
AND final_product.ok = ''

```

ELSE

UPDATE final_product;

SET final_product.ass_date = **CAST**(tx3 AS D), final_product.ok = 'y';

WHERE final_product.pc_id = **CAST**(tx2 AS I) **AND** final_product.ass_id = **CAST**(tx1 AS I)

AND final_product.ok = ''

ENDIF

SELECT final_product.*;

from final_product;

WHERE final_product.ass_id = **CAST**(tx1 AS I) **INTO CURSOR** tmp

Thisform.grid1.recordsource = 'tmp'

5) “calculate income” - вызывает форму позволяющую сборщику рассчитать его приблизительный доход за выбранный период и получит расчет в виде отчета.

Код кнопки:

DO FORM forms/incomecalc

A screenshot of a Windows-style dialog box titled "your id:". It contains three text input fields: "from", "to", and "your id:". The "from" and "to" fields are positioned side-by-side at the top, while the "your id:" field is below them. To the right of the "your id:" field is a button labeled "check out". The dialog box has standard Windows window controls (minimize, maximize, close) in the top right corner.

Для получения расчета пользователь должен ввести промежуток дат за который необходимо произвести расчет и свой id, после чего по нажатию кнопки “check out” будет сформирован отчет.

Код кнопки:

tx1 = thisform.text1.Text

tx2 = thisform.text2.Text

tx3 = thisform.text3.Text

SELECT fp.pc_id as pc_id,fp.ass_date as ass_date,fp.ass_id as ass_id , (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast + mb.coast) as coast;

FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps, motherboard as mb;

WHERE fp.conf_id = com.conf_id **AND** gpu.gpu_id = com.gpu_id **AND** cpu.cpu_id = com.cpu_id **AND** ram. ram_id = com. ram_id **AND** dk.disk_id = com.disk_id **AND** ce.case_id = com.case_id **AND** ps.power_supply_id = com.power_supply_id **AND** mb.motherboard_id = com.motherboard_id **into cursor** tmp

SELECT tmp.*;

from tmp;

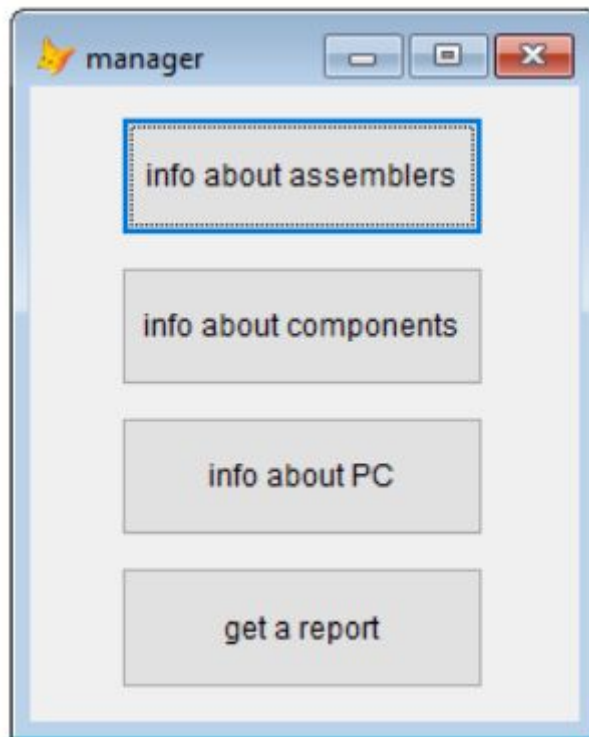
WHERE tmp.ass_id = **CAST**(tx3 AS I) **AND** tmp.ass_date between **CAST**(tx1 AS D) **and** **CAST**(tx2 AS D) **into cursor** tmp1

SELECT tmp1

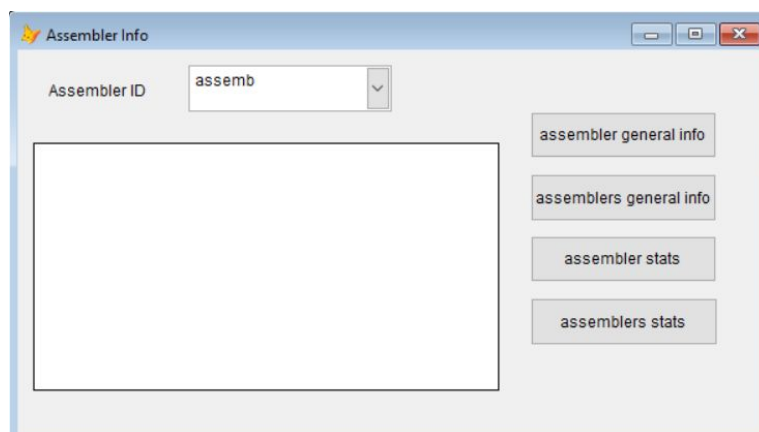
REPORT FORM REPORTS/income preview

Менеджер.

При входе в систему как “менеджер” запустится форма содержащая 4 кнопки:



1) **“info about assemblers”** вызывает форму, в которой можно посмотреть различную информацию о сборщиках.



а) **“assembler general info”** позволяет посмотреть всю основную информацию о выбранном сборщике (сборщик выбирается в combobox assembler ID).

```
assembler = thisform.assemb.Value  
SELECT ass.ass_id, ass.name, ass.marriag_rate FROM assemblers as ass;  
WHERE ass.ass_id = CAST(assembler as I) INTO CURSOR tmp  
Thisform.grid1.columncount=-1  
Thisform.grid1.recordsourcetype=1  
Thisform.grid1.recordsource = 'tmp'
```

б) **“assembler general info”** позволяет всю основную информацию о всех сборщиках

```
SELECT a.* FROM assemblers as a INTO CURSOR tmp4
```

```
Thisform.grid1.columncount=-1
```

```
Thisform.grid1.recordsourcetype=1
```

```
Thisform.grid1.recordsource = 'tmp4'
```

в) “**assembler stats**” позволяет посмотреть кол-во собранных компьютеров, выбранным сборщиком, а также их общая стоимость

```
ssembler = thisform.assemb.value
```

```
SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast + mb.coast) as coast;
```

```
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps, motherboard as mb;
```

```
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO CURSOR tmp_coast
```

```
SELECT ass.ass_id, fp.pc_id, tmp.coast;
```

```
FROM tmp_coast as tmp, assemblers as ass, final_product as fp;
```

```
WHERE fp.pc_id = tmp.pc_id AND fp.ass_id = ass.ass_id INTO CURSOR tmp3
```

```
SELECT tmp3.ass_id, SUM(tmp3.coast) as sum_sell, COUNT(tmp3.pc_id) as count_pc FROM tmp3;
```

```
WHERE tmp3.ass_id = CAST(ssembler as I);
```

```
GROUP BY tmp3.ass_id INTO CURSOR tmp5
```

```
Thisform.grid1.columncount=-1
```

```
Thisform.grid1.recordsourcetype=1
```

```
Thisform.grid1.recordsource = 'tmp5'
```

г) “**assemblers stats**” позволяет посмотреть кол-во собранных компьютеров, всеми сборщиками, а также их общая стоимость

```
ssembler = thisform.assemb.value
```

```
SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast + mb.coast) as coast;
```

```
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps, motherboard as mb;
```

```
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO CURSOR tmp_coast
```

```
SELECT ass.ass_id, fp.pc_id, tmp.coast;
```

```
FROM tmp_coast as tmp, assemblers as ass, final_product as fp;
```

```
WHERE fp.pc_id = tmp.pc_id AND fp.ass_id = ass.ass_id INTO CURSOR tmp3
```

```
SELECT tmp3.ass_id, SUM(tmp3.coast) as sum_sell, COUNT(tmp3.pc_id) as count_pc FROM tmp3;
```

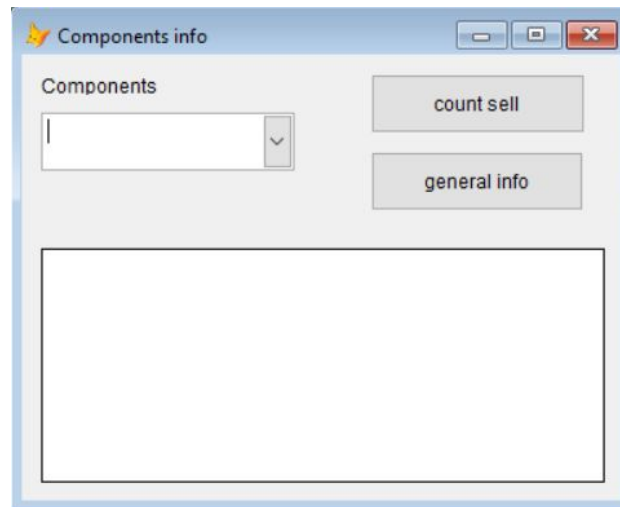
```
GROUP BY tmp3.ass_id INTO CURSOR tmp5
```

```
Thisform.grid1.columncount=-1
```

```
Thisform.grid1.recordsourcetype=1
```

```
Thisform.grid1.recordsource = 'tmp5'
```

2) “**info about components**” вызывает форму, в которой можно посмотреть различную информацию о компонентах.



a) “**count sell**” позволяет посмотреть статистику по продажам среди всех запчастей РС выбранного типа (количество продаж, общая сумма выручки).

```
element = thisform.component.Value
```

```
SELECT fp.conf_id as conf_id, COUNT(fp.conf_id) as count_conf;
```

```
FROM final_product as fp;
```

```
GROUP BY fp.conf_id INTO CURSOR tmp1
```

```
Thisform.grid1.columncount=-1
```

```
Thisform.grid1.recordsourcetype=1
```

DO CASE

CASE element = "gpu"

```
SELECT com.gpu_id as gpu_id, SUM(tmp1.count_conf) as count_sell FROM  
components as com, tmp1;
```

```
WHERE com.conf_id = tmp1.conf_id GROUP BY com.gpu_id INTO CURSOR tmp2
```

```
SELECT g.gpu_id, g.coast, tmp2.count_sell, g.coast*tmp2.count_sell as total_coast
```

```
FROM tmp2, gpu as g;
```

```
WHERE g.gpu_id = tmp2.gpu_id INTO CURSOR tmp
```

```
Thisform.grid1.recordsource = 'tmp'
```

CASE element = "case"

```
SELECT com.case_id as case_id, SUM(tmp1.count_conf) as count_sell FROM  
components as com, tmp1;
```

```
WHERE com.conf_id = tmp1.conf_id GROUP BY com.case_id INTO CURSOR tmp2
```

```
SELECT c.case_id, c.coast, tmp2.count_sell, c.coast*tmp2.count_sell as total_coast
```

```
FROM tmp2, case as c;
```

```

WHERE c.case_id = tmp2.case_id INTO CURSOR tmp
Thisform.grid1.recordsource = 'tmp'
CASE element = "disk"
    SELECT com.disk_id as disk_id, SUM(tmp1.count_conf) as count_sell FROM
components as com, tmp1;
    WHERE com.conf_id = tmp1.conf_id GROUP BY com.disk_id INTO CURSOR tmp2
    SELECT d.disk_id, d.coast, tmp2.count_sell, d.coast*tmp2.count_sell as total_coast
FROM tmp2, disk as d;
    WHERE d.disk_id = tmp2.disk_id INTO CURSOR tmp
    Thisform.grid1.recordsource = 'tmp'
CASE element = "power supply unit"
    SELECT com.power_supply_id as power_supply_id, SUM(tmp1.count_conf) as
count_sell FROM components as com, tmp1;
    WHERE com.conf_id = tmp1.conf_id GROUP BY com.power_supply_id INTO CURSOR
tmp2
    SELECT p.power_supply_id, p.coast, tmp2.count_sell, p.coast*tmp2.count_sell as
total_coast FROM tmp2, power_supply as p;
    WHERE p.power_supply_id = tmp2.power_supply_id INTO CURSOR tmp
    Thisform.grid1.recordsource = 'tmp'
CASE element = "cpu"
    SELECT com.cpu_id as cpu_id, SUM(tmp1.count_conf) as count_sell FROM
components as com, tmp1;
    WHERE com.conf_id = tmp1.conf_id GROUP BY com.cpu_id INTO CURSOR tmp2
    SELECT c.cpu_id, c.coast, tmp2.count_sell, c.coast*tmp2.count_sell as total_coast
FROM tmp2, cpu as c;
    WHERE c.cpu_id = tmp2.cpu_id INTO CURSOR tmp
    Thisform.grid1.recordsource = 'tmp'
CASE element = "ram"
    SELECT com.ram_id as ram_id, SUM(tmp1.count_conf) as count_sell FROM
components as com, tmp1;
    WHERE com.conf_id = tmp1.conf_id GROUP BY com.ram_id INTO CURSOR tmp2
    SELECT r.ram_id, r.coast, tmp2.count_sell, r.coast*tmp2.count_sell as total_coast
FROM tmp2, ram as r;
    WHERE r.ram_id = tmp2.ram_id INTO CURSOR tmp
    Thisform.grid1.recordsource = 'tmp'
CASE element = "motherboard"
    SELECT com.motherboard_id as motherboard_id, SUM(tmp1.count_conf) as count_sell
FROM components as com, tmp1;
    WHERE com.conf_id = tmp1.conf_id GROUP BY com.motherboard_id INTO CURSOR
tmp2
    SELECT m.motherboard_id, m.coast, tmp2.count_sell, m.coast*tmp2.count_sell as
total_coast FROM tmp2, motherboard as m;
    WHERE m.motherboard_id = tmp2.motherboard_id INTO CURSOR tmp
    Thisform.grid1.recordsource = 'tmp'
ENDCASE

```

б) “general info” показывает общую информацию по о запчастях выбранного типа.

```

element = thisform.component.Value
Thisform.grid1.columncount=-1
Thisform.grid1.recordsourcetype=1
DO CASE
    CASE element = "gpu"
        SELECT g.* FROM gpu as g INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "case"
        SELECT c.* FROM case as c INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "disk"
        SELECT d.* FROM disk as d INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "power supply unit"
        SELECT p.* FROM power_supply as p INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "cpu"
        SELECT c.* FROM cpu as c INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "ram"
        SELECT r.* FROM ram as r INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
    CASE element = "motherboard"
        SELECT m.* FROM motherboard as m INTO CURSOR tmp
        Thisform.grid1.recordsource = 'tmp'
ENDCASE

```

3) **“info about PC”** позволяет посмотреть различную информацию о заказах, которые компания уже выполнила или собирается выполнить.

a) **“view interval cash”** выдает информацию о выполненных заказах за выбранный период.

```

date1 = thisform.text1.Text
date2 = thisform.text2.Text

```



```

SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast +
mb.coast) as coast;
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,
motherboard as mb;
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO
CURSOR tmp
SELECT fp.pc_id, fp.ass_id, fp.ok, tmp.coast, fp.ass_date;
FROM final_product as fp, tmp;
WHERE tmp.pc_id=fp.pc_id AND fp.ass_id <> 0 AND fp.ass_date between CAST(date1 as D) AND
CAST(date2 as D) INTO CURSOR tmp2
Thisform.grid1.recordsource = 'tmp2'

```

б) “view interval total cash” выдает общее количество средств, которое компания получила за данный период.

```

date1 = thisform.text1.Text
date2 = thisform.text2.Text
SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast +
mb.coast) as coast;
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,
motherboard as mb;
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO
CURSOR tmp
SELECT SUM(tmp.coast);
FROM final_product as fp, tmp;
WHERE tmp.pc_id=fp.pc_id AND fp.ass_id <> 0 AND fp.ass_date between CAST(date1 as D) AND
CAST(date2 as D) INTO CURSOR tmp2
Thisform.grid1.recordsource = 'tmp2'

```

в) “total cash” выдает количество денег, которое компания получила в целом.

```

SELECT (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast + mb.coast) as coast;
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,
motherboard as mb;
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO
CURSOR tmp
SELECT SUM(tmp.coast) FROM tmp into CURSOR tmp
Thisform.grid1.recordsource = 'tmp'

```

д) “free projects” показывает свободные проекты (которые еще никто не собирает).

```

SELECT final_product.pc_id, final_product.ass_id FROM final_product;
WHERE final_product.ass_id = 0 INTO CURSOR tmp

```

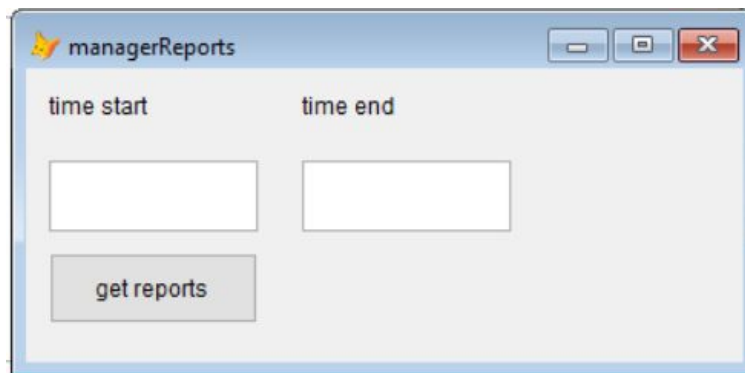
```
Thisform.grid1.recordsource = 'tmp'
```

e) **“general information”** дает всю информацию по всем заказам.

```
SELECT fp.* FROM final_product as fp INTO CURSOR tmp
```

```
Thisform.grid1.recordsource = 'tmp'
```

4) **“get a report”** вызывает форму в которой можно сгенерировать отчет о прибыли компании за заданный период.



```
date1 = thisform.text1.Text
```

```
date2 = thisform.text2.Text
```

```
SELECT fp.pc_id as pc_id, (gpu.coast + cpu.coast + ram.coast + dk.coast + ce.coast + ps.coast +  
mb.coast) as coast;
```

```
FROM final_product as fp, components as com, gpu, cpu, ram, disk as dk, case as ce, power_supply as ps,  
motherboard as mb;
```

```
WHERE fp.conf_id = com.conf_id AND gpu.gpu_id = com.gpu_id AND cpu.cpu_id = com.cpu_id AND  
ram.ram_id = com.ram_id AND dk.disk_id = com.disk_id AND ce.case_id = com.case_id AND  
ps.power_supply_id = com.power_supply_id AND mb.motherboard_id = com.motherboard_id INTO  
CURSOR tmp
```

```
SELECT fp.pc_id as pc, fp.ass_id as assembler, fp.ok as OK, tmp.coast as coast, fp.ass_date as  
asb_date;
```

```
FROM final_product as fp, tmp;
```

```
WHERE tmp.pc_id=fp.pc_id AND fp.ass_id <> 0 AND fp.ass_date between CAST(date1 as D) AND  
CAST(date2 as D) INTO CURSOR tmp2
```

```
SELECT tmp2.assembler as assembler, SUM(tmp2.coast*1.1) as coast;
```

```
FROM tmp2;
```

```
GROUP BY tmp2.assembler INTO CURSOR tmp3
```

```
SELECT tmp3.assembler, tmp3.coast, asmb.marriag_rate;
```

```
FROM assemblers as asmb, tmp3;
```

```
WHERE asmb.ass_id = tmp3.assembler INTO CURSOR tmp10
```

```
SELECT tmp10
```

```
REPORT FORM REPORTS/reportmanager PREVIEW
```

3.2.2 Отложенная программа (отчеты)

У покупателя отчет - это чек. В чеке содержится информация о компонентах компьютера (id и стоимость каждой детали), плата за услуги сборки, дата сборки и итоговая сумма. Пример:

cheque			
12/15/20			
Case id and coast	121	40\$	
RAM id and coast	133600	20\$	
Power_supply id and coast	105501	60\$	
Disk id and coast	115121	35\$	
Motherboard id and coast	14133665	35\$	
GPU id and coast	159801	250\$	
CPU id and coast	161336	200\$	
Fee for assembly	96.00	\$	
The build date 10/28/20			
Total coast 736.00 \$			

Для сборщика отчетом является форма содержащая отчет о проделанной им работе за выбранный период, на основе которой он может рассчитать свою заработную плату. Отчет содержит id собранных за выбранный период компьютеров, их стоимость и часть наценки, которую получает сборщик. Пример отчета:

Your income for the selected period			12/15/20
pc_id	cost:	my income:	
1	945	47.25	
13	945	47.25	
33	885	44.25	
9	2354	117.70	
your income wage is 256.45			net of tax of 30%

Отчетом менеджера служит доходность за выбранный период, на основе которой можно получить всю необходимую информацию для выработки стратегии дальнейшего управления. Отчет содержит ID каждого сборщика, деньги, которые он принес компании, его рейтинг, а также весь заработок за этот период. Пример отчета:

Income statement		
assembler:	1	
income:	1186.9	
rate:	7	
assembler:	2	
income:	3428.9	
rate:	10	
assembler:	3	
income:	704.0	
rate:	8	
assembler:	4	
income:	818.4	
rate:	6	
assembler:	5	
income:	8949.6	
rate:	10	
assembler:	7	
income:	1164.9	
rate:	1	
Total income: 16452.7		

Работу выполнили:

Мукин Юрий Дмитриевич - реализация клиентского приложения сборщика, сборка проекта.

Хахин Максим Сергеевич - реализация клиентского приложения покупателя, заполнение таблиц.

Сыроежкин Кирилл Геннадьевич - реализация клиентского приложения менеджера, написание технического задания.