МИНЕСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ (МГТУ ГА)

Кафедра вычислительных машин, комплексов, систем и сетей

Практическая работа выполнена

с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя, дата)

Практическая работа по дисциплине

"БАЗЫ ДАННЫХ"

**Тема**

«Система штурманского обеспечения полётов»

Работу выполнил: Студент группы ЭВМ 3-2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Новичков В.Д.

(Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

Руководитель:   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вайнейкис Л.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_доцент, к.т.н (степень, звание, Ф.И.О)

Москва 2013г.

# Тема.

Разработать систему штурманского обеспечения полётов

# Исходные данные

**Система содержит:**

1. Данные о промежуточных пунктах маршрута (Название, номер ППМ, координаты, страна, город)
2. Данные о самолёте (Тип самолёта, бортовой номер, скорость взлёта, посадки и крейсерную скорость, взлётную массу)
3. Список трасс (код трассы, название)
4. Список ППМ на маршруте (Код трассы, Код ППМ, порядковый номер ППМ)

**Система должна выполнять следующие функции:**

1. Выводить информацию об аэропортах
2. Выводить план полёта, состоящей из списка ППМ.
3. Рассчитывать время полёта и расстояние между ППМ, а так же вывести общее время полёта и пройденное самолётом расстояние за весь полёт.
4. Система должна иметь средство ввода и обновления данных в базе данных

# Введение

Базы данных - совокупность данных, организованная по определенным правилам, предусматривающая общие принципы описания, хранения, манипулирования данными, независимыми от прикладных программ.

СУБД – система управления базами данных – совокупность программ, предназначенных для управления БД и возможности получения пользователями необходимой информации из базы. В задачи СУБД входят следующие задачи:

* Формирование и поддержание БД
* Обработка информации
* Прием запросов
* Предоставление информации пользователям
* Обеспечение целостности и реорганизации ценностей БД
* Организация совместной работы пользователей

На сегодняшний день существует множество различных систем управления базами данных. Они все используют разные средства и функции, но преимущественно у всех СУБД в основе лежат одинаковые понятия. Поэтому для обобщения этих понятий, приемов и методов на весь класс СУБД, я хотела бы взять программу, входящую в Microsoft Office, Microsoft Access.

Microsoft Access –реляционная СУБД, в которой предусмотрены все необходимые средства для определения и обработки данных, а также управления ими при работе с большим объемом информации.

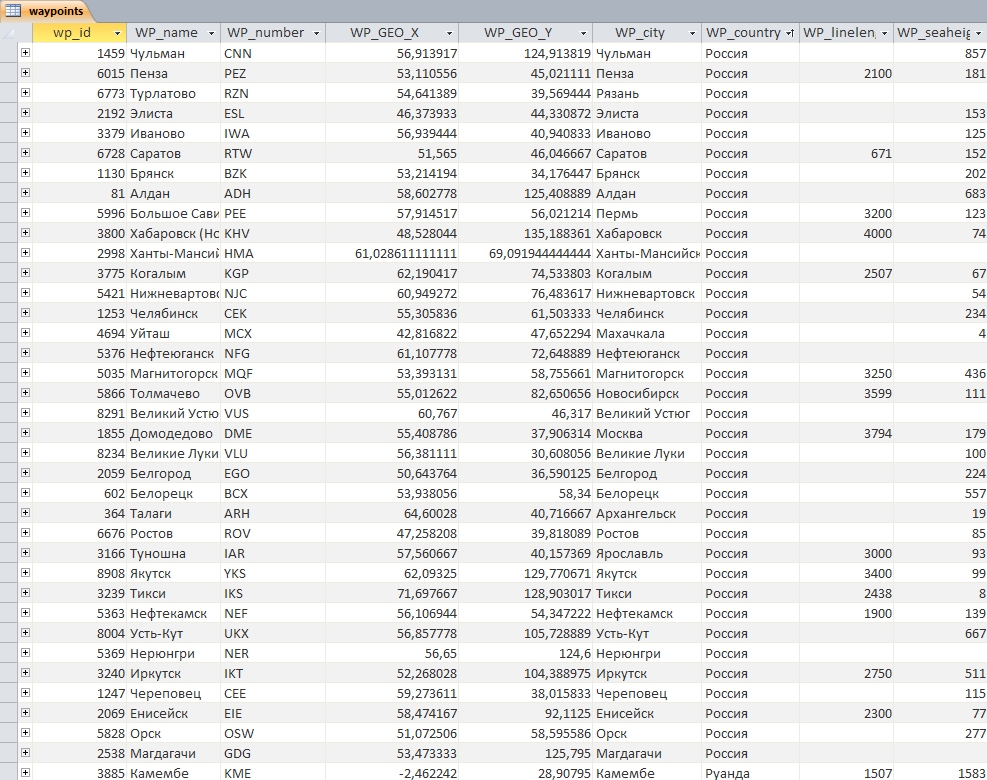
Access - функционально полная система, имеющая мощные средства для работы в этой программе. Ее преимуществом перед другими является простота, наличие всех средств для успешной обработки и управления БД.

# Структура Базы данных

## “Система штурманского обеспечения полёта”

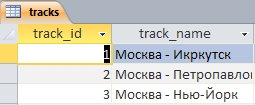
1. **Аэропорты мира:**

WP\_id(integer, Primary key), WP\_name(VARCHAR), WP\_number(VARCHAR), WP\_GEO\_X (Double), WP\_GEO\_Y(Double), WP\_city, WP\_country, WP\_linelenght(double), WP\_seaheight(double)



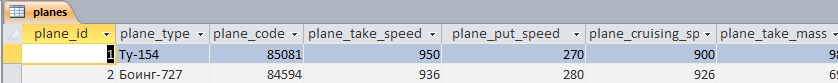
1. **Трассы**

trach\_id (integer, Primary key), track\_name(varchar)



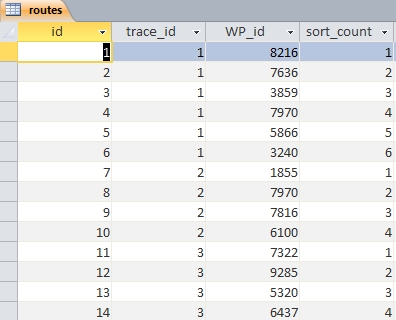
1. **Самолёты**

plane\_id(integer, primary key), plany\_type(varchar), plane\_code(int), plane\_take\_speed(double), plane\_put\_speed(double), plane\_cruising\_speed(double), plane\_take\_mass(double)

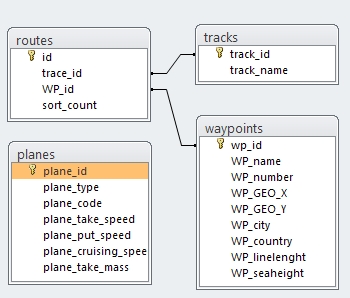


1. **Планы полётов**

Id(int,primary key), trace\_id(int), WP\_id(int), sort\_count(int)

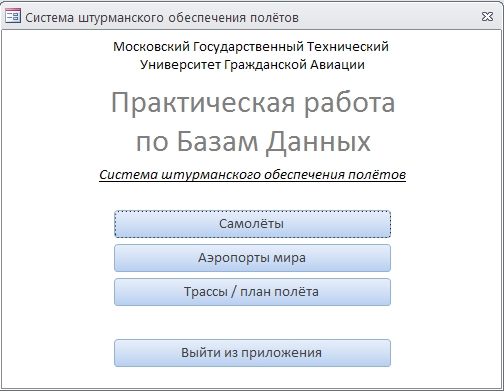
****

**Структура базы данных:**



# Создание пользовательского интерфейса

## Главное меню



Option Compare Database

Private Sub Form\_Open(Cancel As Integer)

' Minimize the database window and initialize the form.

On Error GoTo Form\_Open\_Err

' Minimize the database window.

DoCmd.SelectObject acForm, "title", True

DoCmd.Minimize

' Move to the switchboard page that is marked as the default.

Me.Filter = "[ItemNumber] = 0 AND [Argument] = 'Default' "

Me.FilterOn = True

Form\_Open\_Exit:

Exit Sub

Form\_Open\_Err:

MsgBox Err.Description

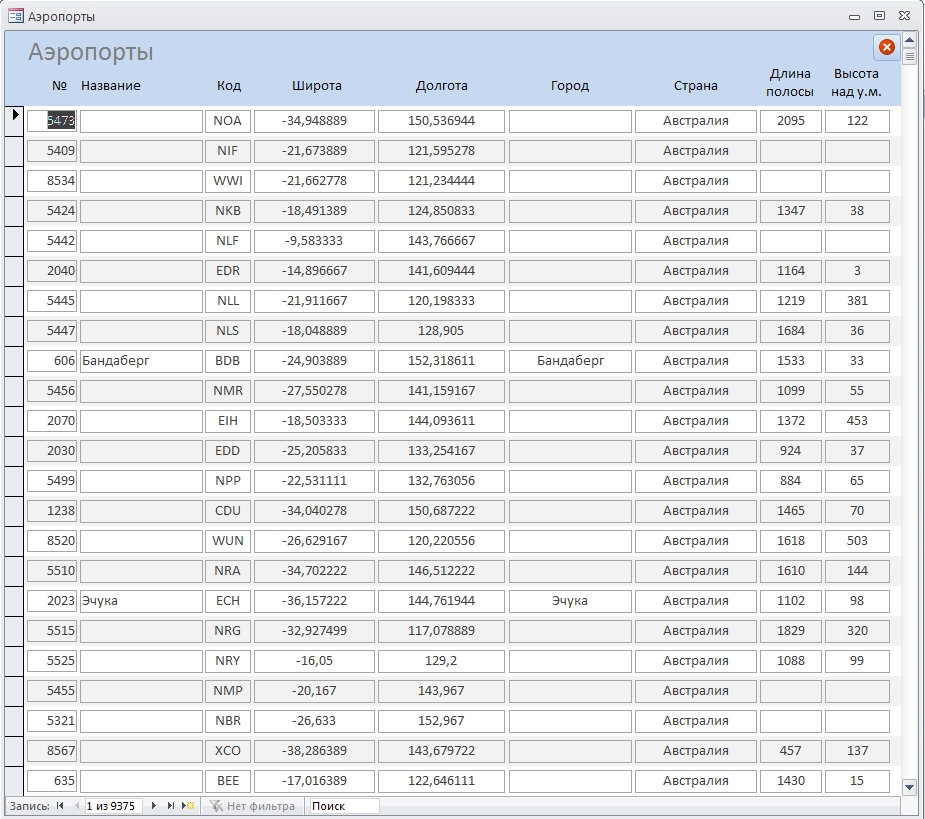
Resume Form\_Open\_Exit

End Sub

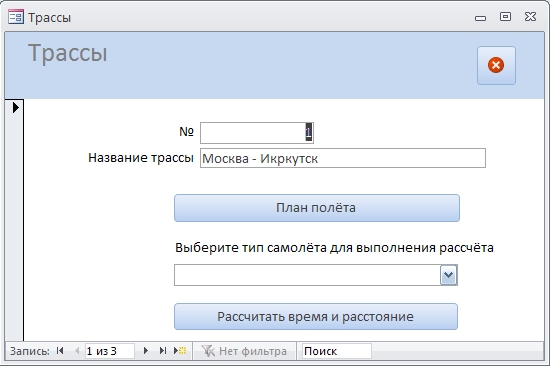
## Список самолётов

## Y:\Документы\Мои рисунки\Скриншоты\Scr 2013.06.27 05-23-12-057.jpg

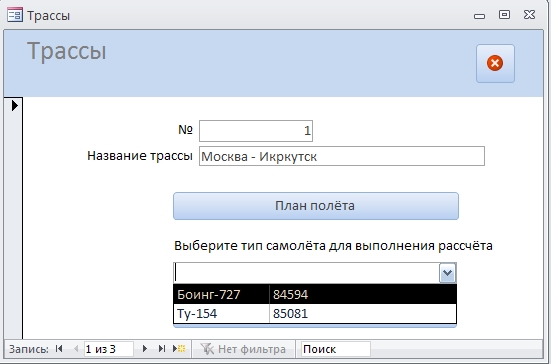
## Список аэропортов мира – ППМ



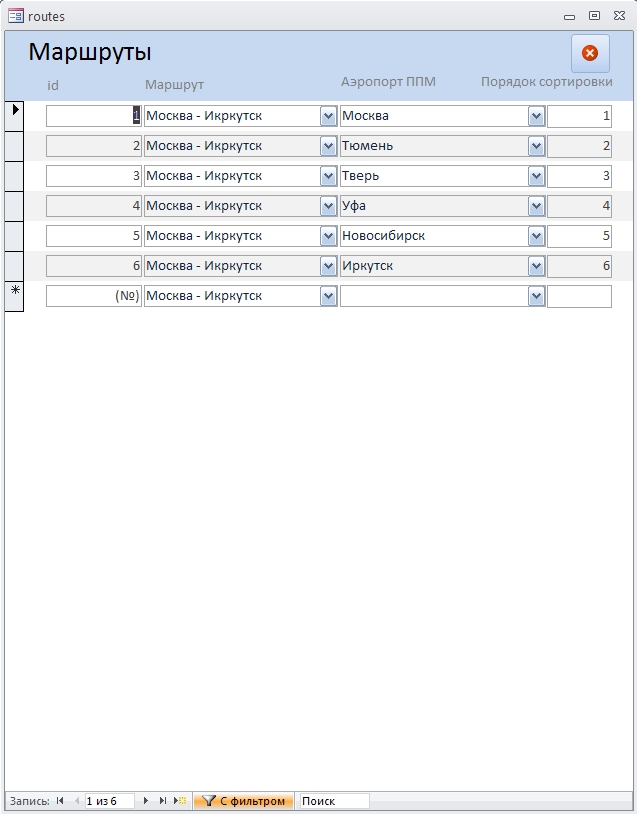
## Список трасс



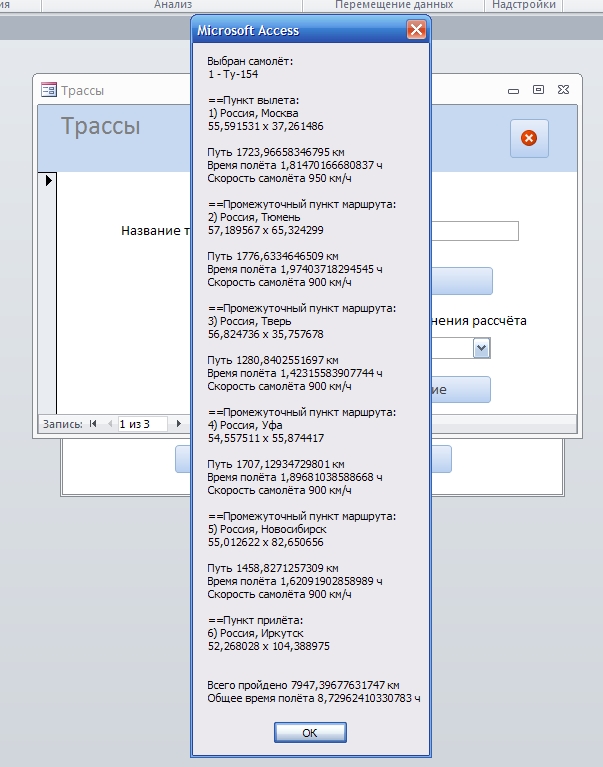
Выбор самолёта из комбо-бокса: (данные из таблицы “самолёты”)



## Просмотр/редактирование плана полёта



## Рассчёт времени полёта и расстояния по плану полёта для выбранного типа самолёта



## Исходный код формы:

Option Compare Database

Private Sub waypointsButton\_Click()

End Sub

Private Sub Кнопка8\_Click()

Dim rs As Recordset, rs1() As String, rs2 As Recordset, Routes As Recordset, RtrCnt As Recordset, \_

TstStr As String, TheLenght As Double, count As Integer, TimeOfFly As Double, TimeOfFly\_full As Double, \_

PlaneSpeed As Double

If IsNull(plane.Value) Then

MsgBox "Самолёт не выбран!"

Exit Sub

End If

Set rs = CurrentDb.OpenRecordset("SELECT \* FROM planes WHERE plane\_id = " & plane.Value & ";")

TheTest = "Выбран самолёт:" & Chr(13) & rs("plane\_id") & " - " & rs("plane\_type") & Chr(13) & Chr(13)

Set Routes = CurrentDb.OpenRecordset("SELECT \* FROM routes WHERE trace\_id = " & \_

track\_id.Value & " ORDER BY sort\_count;")

Set RtrCnt = CurrentDb.OpenRecordset("SELECT count(\*) FROM routes WHERE trace\_id = " & \_

track\_id.Value & ";")

Dim \_

WP\_Geo\_x As Double, \_

WP\_Geo\_y As Double

count = 0

TheLenght = 0

TheLngt = 0

PlaneSpeed = 0

TimeOfFly = 0

TimeOfFly\_full = 0

Do While Not Routes.EOF

Set rs2 = CurrentDb.OpenRecordset("SELECT \* FROM waypoints WHERE WP\_id = " \_

& Routes("WP\_id") & ";")

If (count = 0) Then

" & WP\_id\_1.Value & ";")

TheTest = TheTest & ForTest.WayPointInfo(rs2, "==Пункт вылета:", count + 1)

WP\_Geo\_x = rs2("WP\_GEO\_X")

WP\_Geo\_y = rs2("WP\_GEO\_Y")

Else

TheLngt = ForTest.VincentyDistance(WP\_Geo\_x, rs2("WP\_GEO\_X"), \_

WP\_Geo\_y, rs2("WP\_GEO\_Y"))

TheLenght = TheLenght + TheLngt

If (count = 1) Then

PlaneSpeed = rs("plane\_take\_speed")

TimeOfFly = TheLngt / PlaneSpeed

Else

If (count = (RtrCnt(0) - 1)) Then

PlaneSpeed = rs("plane\_cruising\_speed")

'PlaneSpeed = rs("plane\_put\_speed")

TimeOfFly = TheLngt / PlaneSpeed

Else

PlaneSpeed = rs("plane\_cruising\_speed")

TimeOfFly = TheLngt / PlaneSpeed

End If

End If

TimeOfFly\_full = TimeOfFly\_full + TimeOfFly

WP\_Geo\_x = rs2("WP\_GEO\_X")

WP\_Geo\_y = rs2("WP\_GEO\_Y")

TheTest = TheTest & "Путь " & TheLngt & " км" & Chr(13)

TheTest = TheTest & "Время полёта " & TimeOfFly & " ч" & Chr(13)

TheTest = TheTest & "Скорость самолёта " & PlaneSpeed & " км/ч" & \_

Chr(13) & Chr(13)

If (count = (RtrCnt(0) - 1)) Then

TheTest = TheTest & ForTest.WayPointInfo(rs2, "==Пункт прилёта:", \_

count + 1)

Else

TheTest = TheTest & ForTest.WayPointInfo(rs2, \_

"==Промежуточный пункт маршрута:", count + 1)

End If

End If

count = count + 1

Routes.MoveNext

Loop

MsgBox TheTest & Chr(13) & "Всего пройдено " & TheLenght & " км" & \_

Chr(13) & "Общее время полёта " & TimeOfFly\_full & " ч"

End Sub

## Исходный код модуля “ForTest ”

Option Compare Database

Public Function test() As String

test = "Вася пупкин отжигает!=)))))"

End Function

Public Function WayPointInfo(wp\_n, txt, cnt) As String

WayPointInfo = txt & Chr(13) & cnt & ") " & wp\_n("WP\_country") & ", " & wp\_n("WP\_city") & Chr(13) & wp\_n("WP\_GEO\_X") & " x " & wp\_n("WP\_GEO\_Y") & Chr(13) & Chr(13)

End Function

Function ArcCos(X As Double) As Double

If Abs(X) > 1# Then X = X - Fix(X)

ArcCos = Atn(-X / Sqr(-X \* X + 1)) + 2 \* Atn(1)

End Function

Function ArcSin(X As Double) As Double

If Abs(X) > 1# Then X = X - Fix(X)

ArcSin = Atn(X / Sqr(-X \* X + 1))

End Function

Public Function VincentyDistance(SourceLatDeg, DestLatDeg, \_

SourceLongDeg, DestLongDeg) As Double

Const EarthRadiusKM As Double = 6371.01 'Kilometers

Const EarthRadiusSM As Double = 3958.76 'Statute Miles

Const EarthRadiusNM As Double = 3440.07 'Nautical Miles

Const Pi As Double = 3.14159265358979

Const Epsilon As Double = 0.000000000001

Dim EarthRadius As Double

Dim SourceLatRad As Double

Dim SourceLongRad As Double

Dim DestLatRad As Double

Dim DestLongRad As Double

Dim DeltaLongRad As Double

Dim Numerator1 As Double

Dim Numerator2 As Double

Dim Numerator As Double

Dim Denominator As Double

EarthRadius = EarthRadiusKM

SourceLatRad = DegrToRad(SourceLatDeg)

SourceLongRad = DegrToRad(SourceLongDeg)

DestLatRad = DegrToRad(DestLatDeg)

DestLongRad = DegrToRad(DestLongDeg)

DeltaLongRad = Abs(SourceLongRad - DestLongRad)

Numerator1 = Cos(DestLatRad) \* Sin(DeltaLongRad)

Numerator1 = Numerator1 \* Numerator1

Numerator2 = Cos(SourceLatRad) \* Sin(DestLatRad)

Numerator2 = Numerator2 - Sin(SourceLatRad) \* Cos(DestLatRad) \* Cos(DeltaLongRad)

Numerator2 = Numerator2 \* Numerator2

Numerator = Numerator1 + Numerator2

Numerator = Sqr(Numerator)

Denominator = Sin(SourceLatRad) \* Sin(DestLatRad)

Denominator = Denominator + Cos(SourceLatRad) \* Cos(DestLatRad) \* Cos(DeltaLongRad)

If Abs(Denominator) < Epsilon Then

VincentyDistance = Pi / 2#

Else

VincentyDistance = Numerator / Denominator

VincentyDistance = Atn(VincentyDistance)

End If

VincentyDistance = VincentyDistance \* EarthRadius

End Function

Public Function DegToRad(Degrees As Double, Minutes As Double, Seconds As Double) As Double

Const Pi As Double = 3.14159265358979

Degrees = Degrees + Minutes / 60#

Degrees = Degrees + Seconds / 3600#

DegToRad = Degrees \* Pi / 180#

End Function

Public Function DegrToRad(Degrees) As Double

Const Pi As Double = 3.14159265358979

DegrToRad = Degrees \* Pi / 180

End Function