Урок 1

План заняття:

1. Введення. Поняття про БД, СУБД. Типи БД
2. 12 правил Кодда, яким повинна відповідати реляційна СУБД
3. Початок роботи з Microsoft Access. Переваги та недоліки
4. Проектування бази даних
5. Створення бази даних
6. Складові частини бази даних MS Access
7. Режими роботи MS Access
8. Таблиці. Порядок та методи створення таблиць в MS Access
9. Типи полів
10. Домашнє завдання

1.Введення. Поняття про БД, СУБД. Типи БД

Сьогодні ми розпочинаємо вивчення нового курсу «Бази даних». З даним поняттям Ви вже знайомі, оскільки воно вже вжилось в наше сьогодення. Практично кожен з Вас користується базою даних кожен день, хоча, можливо, цього і не помічає. Прикладом баз даних можуть слугувати записна книжка, телефонний довідник, енциклопедії, бібліотечних каталог тощо, в яких зберігається певна необхідна інформація. Так, в телефонному довіднику міститься інформація про людей: їх імена, прізвища, телефони, адреси тощо. В енциклопедіях – дані по певній предметній області.

Отже, підсумовуючи дане можна сказати, що **база даних (БД)** – це сукупність даних, пов'язаних між собою певною тематикою. Ці дані зберігається на машинних носіях системи в впорядкованому вигляді.

Поряд із поняттям баз даних існує поняття **системи управляння базами даних (СУБД)** – це комп'ютерна програма, яка дозволяє управляти базою даних, тобто створювати і працювати з нею.

Зазвичай сучасні СУБД містять **наступні складові**:

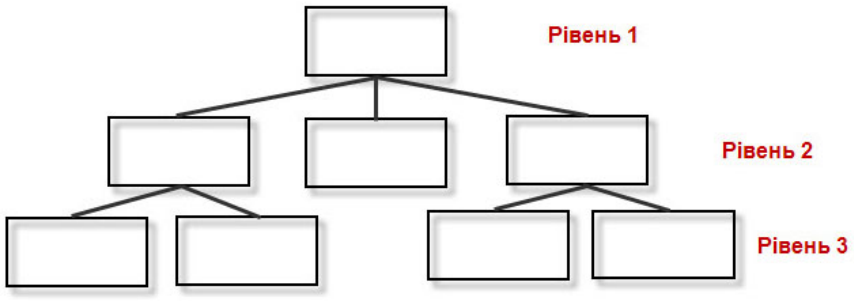
- ядро СУБД, яке відповідає за управління даними і їх журналізацію;
- процесор мови баз даних, який забезпечує оптимізацію запитів і переведенням їх на машинну мову;
- підсистему підтримки часу виконання (runtime), яка створює користувацький інтерфейс і забезпечує його взаємозв'язок з базою даних;
- сервісні програми, тобто набір утиліт, які надають додаткові можливості по обслуговуванню інформаційної системи.

В сучасних комп'ютерних інформаційних системах найчастіше застосовуються 4 **типи моделей БД і СУБД:**

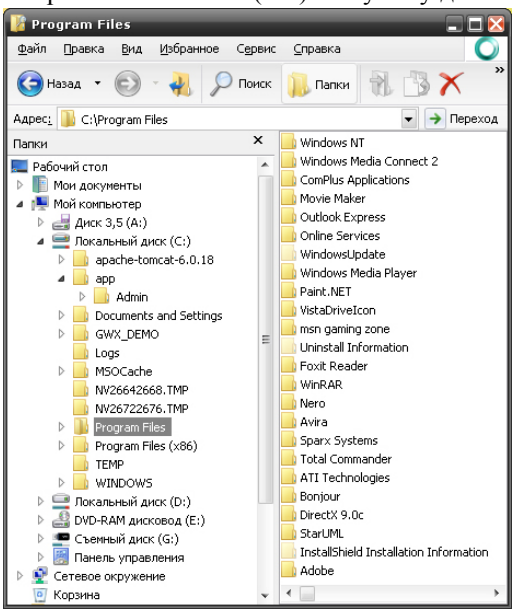
1. ієрархічні;
2. сіткові;
3. реляційні;
4. об'єктно-орієнтовані.

Ієрархічні бази даних. Це самий перший вид баз даних, використовуваний мирним населенням (☺). Таку базу даних графічно можна представити у вигляді дерева, яке складається з об'єктів різних рівнів. Верхній рівень займає один об'єкт, другий – об'єкти другого рівня і так далі. Між об'єктами існують зв'язки «батько-потомок» і тому кожен об'єкт може включати в себе кілька об'єктів більш низького рівня.

Схематично така модель буде виглядати наступним чином:



Типовим прикладом ієрархічної бази даних являється каталог папок Windows – «Провідник». Верхній рівень дерева займає «Робочий стіл», на другому рівні знаходяться папки «Мій комп'ютер», «Мої документи»,



«Сетевое окружение» і «Корзина», які являються дочірними по відношенню до Робочого столу. Далі розміщуються об'єкти третього рівня тощо.

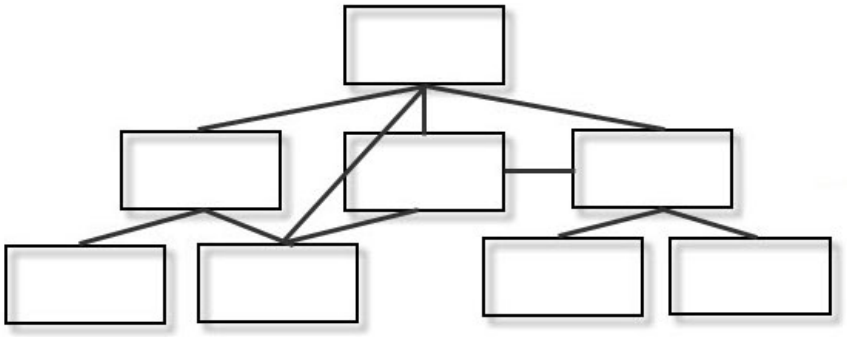
Наведемо ще ряд **прикладів ієрархічних баз даних і СУБД**:

- ✓ реєстр Windows;
- ✓ сервери каталогів, такі як LDAP Active та Directory;
- ✓ System-2000 від міжнародної компанії SAS-Institute;
- ✓ Information Management System (IMS) фірми IBM.

Основним **недоліком** таких баз даних є те, що у випадку, якщо дані не мають такої структури, тобто їх не можна представити у вигляді дерева, виникає багато складнощів при побудові бази даних. Причому така база даних буде непродуктивною.

До недоліків слід віднести і той факт, що у випадку пошуку даних не можна направити потік пошуку знизу вверх.

Сіткові бази даних. Являється узальнюючою моделлю ієрархічної. Кожен об'єкт в такій базі даних може мати більше одного батьківського елемента. Простішими словами в сітковій базі даних, елементи можуть бути зв'язані між собою довільним чином. Схематичне зображення даної моделі представлене нижче:

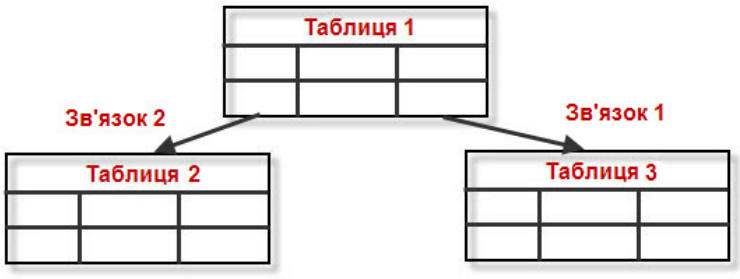


Характерним прикладом такої бази даних є Глобальна мережа Інтернет, в якій гіперпосилання зв'язують між собою сотні мільонів документів. Всі ці зв'язані документи являють собою єдину розприділену сіткову базу даних.

Прикладами сіткової моделі можуть слугувати такі бази даних як СООБЗ (сіткова об'єктно-орієнтована база знань) Cerebrum та CronosPRO.

Недоліком такої моделі являється низька продуктивність, яка призводить до того, що навіть прості запити виконуються досить складно. Крім того, при роботі з такою базою даних, прикладний програміст повинен знати багато термінів, вивчити декілька внутрішніх мов баз даних, детально уявляти логічну структуру бази даних для здійснення навігації серед різних екземплярів, наборів, записів тощо.

Реляційні (табличні) бази даних. Слово «реляційа» походить від англійського «relation» (зв'язок). В такій моделі, всі дані в базі даних зберігаються у вигляді двохвимірних таблиць, що складаються з стовпчиків та рядків. Кожна таблиця містить сукупність даних одного типу, які мають однакові властивості. Стовпчики такої таблиці називаю полями або атрибутами та містять інформацію про властивості об'єкта, а рядки (записи, кортежи) містять значення конкретної властивості.

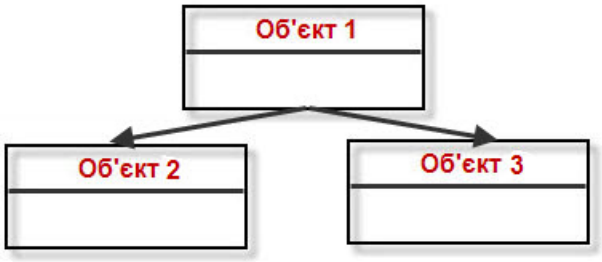


Приклади реляційних СУБД:

- ✓ Microsoft SQL Server;
- ✓ Oracle;
- ✓ Interbase/Firebird;
- ✓ Microsoft Access;
- ✓ MySQL.

Основна перевага цієї моделі даних – це простота та наочність бази даних при її проектуванні. Саме тому вона є самою розповсюдженою моделлю БД і саме на такій моделі ми і зупинимось при вивченні даного курсу.

Об'єктно-орієнтовані бази даних. Дані в такій моделі бази даних оформлені у вигляді моделей об'єктів, які використовують прикладні програми, що управляють зовнішніми подіями. Варто відмітити, що ряд об'єктно-орієнтованих баз даних розроблені для тісної взаємодії з об'єктно-орієнтованими мовами програмування (C++, C#, Java, Python, Visual Basic .NET, Smalltalk тощо) і використовують таку ж саму модель, що і останні. Такі бази даних зазвичай рекомендовані для тих випадків, коли вимагається високопродуктивна обробка даних, що мають складну структуру.



Об'єктно-орієнтовані бази даних повинні відповідати двом **основним критеріям**: система повинна бути об'єктно-орієнтованою і являти собою базу даних. Фактично така модель поєднує в собі характеристики ОО мов програмування (успадкування, інкапсуляцію, поліморфізм) і баз даних (цілісність даних, безпека, стабільність, транзакції, архівування, відновлення з резервної копії, запити тощо).

Приклади об'єктно-орієнтованих СУБД:

- ✓ IBM Lotus Notes/Domino;
- ✓ Jasmine;
- ✓ ObjectStore;
- ✓ db4objects;
- ✓ ODB-Jupiter.

2.12 правил Кодда, яким повинна відповідати реляційна СУБД

Оскільки реляційна модель баз даних являється найпоширенішою за рахунок простоти побудови і роботи з базою даних, то саме на на ній ми і зосередимо свою увагу. Але, перед тим, як розпочати засвоєння практичних навичок, потрібно засвоїти основні правила, яким повинна відповідати будь-яка реляційна база даних. Ці правила були запропоновані у 1970р. Англійським математиком Е.Ф. Коддом з компанії IBM.

В термінології Кодда таблиці реляційної бази даних називаються **відношеннями (relation)**, - звідси і назва самої моделі. Всіх правил – 12 (насправді 13). Основне правило, яке не входить до переліку, говорить про те, що **будь-яка реляційна СУБД повинна повністю управляти базою даних, використовуючи зв'язки між даними**.

Всі інші 12 правил представлені нижче:

1. **Правило інформації** (The Information Rule). Вся інформація в базі даних повинна бути представлена у вигляді даних, що зберігаються в комірках таблиць. Причому порядок записів (рядків) в реляційній таблиці не повинен впливати на суть зберігаємих даних.

2. **Правило гарантованого доступу** (Guaranteed Access Rule). Доступ до даних в реляційній БД повинен забезпечуватись шляхом використання комбінації імені таблиці, первинного ключа та імені стовпчика.

3. **Правило підтримки недійсних значень** (Systematic Treatment of Null Values). Вказує на те, що будь-які відсутні дані можна представити за допомогою недійсного значення NULL.

4. **Правило інтерактивного каталога, основаного на реляційній моделі** (Active On-Line Catalog Based on the Relational Model). Вказує на те, що реляційна база даних повинна сама себе описувати. Іншими словами, база данних повинна містити набір реляційних системних таблиць, що описують структуру самої бази даних. При цьому, потрібно забезпечити, щоб доступ до цих таблиць був аналогічний доступу до звичайних таблиць бази даних.

5. **Правило вичерпної підмови даних** (Comprehensive Data Sublanguage Rule). Реляційна СУБД повинна підтримувати хоча б одну реляційну мову, яка має:
   - лінійний синтаксис;
   - може використовуватись як інтерактивно, так і для прикладних додатків;
   - підтримує операції роботи з даними (вставка, видалення, обновлення), забезпечує їх цілісність тощо.

6. **Можливість модифікації представлень** (View Updating Rule). Всі представлення (view) повинні підтримувати операції маніпулювання даними, подібно реляційним таблицям: вибірка даних, вставка, обновлення та їх видалення.

7. **Правило високорівневих операцій додавання, обновлення і видалення** (High-Level Insert, Update, and Delete). Тут акцентується увага на тому, що бази даних по своїй природі орієнтовані на множини. Воно вимагає, щоб операції додавання, видалення та обновлення можна було виконувати як над одиничними даними, так і над множинами записів (рядків).

8. **Правило незалежності фізичних даних** (Physical Data Independence). Вказує на те, що дані бази даних не повинні залежати від використовуваних способів збереження даних на носіях, від апаратного забезпечення комп'ютера, на яких встановлена СУБД.

9. **Правило незалежності логічних даних** (Logical Data Independence). Представлення даних в прикладних додатках не повинно залежати від структури реляційних таблиць. Іншими словами, якщо в базі даних відбулась зміна її структури (деякі таблиці були розділені на кілька, добавлені інші таблиці тощо), то в прикладній програмі відображення даних не повинно змінитись.

10. **Правило незалежності контроля цілісності** (Integrity Independence). Вся інформація, яка необхідна для забезпечення цілісності даних повинна зберігатись в системних таблицях. Використовуючи ці дані, СУБД повинна перевіряти всі вхідні дані на відповідність їх умовам цілісності.

11. **Правило незалежності розприділення** (Distribution Independence). Дозволяє базі даних бути розприділеною, тобто знаходитись на різних комп'ютерах. Причому перенесення даних на інший комп'ютер не повинно вплинути на її роботоздатність.

12. **Правило узгодженості мов різних рівней** (The Nonsubversion Rule). Якщо використовується низькорівнева мова для доступу до даних, то вона не повинна ігнорувати правила безпеки і цілісності даних, які підтримуються мовою більш низького рівня.

Варто відмітити, що насправді, через суворість даних правил, не всі сучасні реляційні СУБД їх дотримуються.

3.Початок роботи з Microsoft Access. Переваги та недоліки

Однією з самих наочних і простих реляційних СУБД на сьогоднішній день являється СУБД Microsoft Access, яка входить до складу пакета продуктів Microsoft Office. Не вдаючись в лабіринти історії створення даної СУБД, розглянемо її переваги та недоліки, а також основні засади її будови.

Основна **перевага** MS Access – це **візуальність**. Засобами даної СУБД можла легко переглянути структуру бази даних, вміст таблиць, зручно і швидко здійснити модифікацію даних тощо. Однак, MS Accesss являється desktop-додатком і тому, стає неможливим організація клієнт-серверної взаємодії прикладної програми і бази даних. Це основний **недолік** даної СУБД. До суттєвих недоліків MS Access також слід віднести:

- неможливість збереження даних великого об'єму;
- низький рівень безпеки;
- неповноцінна підтримка мови даних (SQL).

Фактично MS Access призначений для персонального використання, а у випадку необхідності розробки бази даних, наприклад, великого підприємства, краще скористатись серверною СУБД, наприклад, Oracle, MS SQL Server, MySQL тощо.

Як Ви знаєте, сама по собі база даних - це файл (або група файлів), записаних в спеціальному форматі. Диспетчером даних в MS Access, що виконує завантаження і збереження даних в базі даних користувача, являється ядро бази даних **Microsoft Jet**. Починаючи з MS Access 2000 використовувалось ядро Microsoft Jet версії 4.0. Ця версія Jet підтримує двухбайтове представлення символів (формат Unicode), покращена підтримка мови SQL специфікації ANSI SQL 92 (але вона ще далека від класичної реалізації SQL), реалізована вбудована підтримка інтерфейсів OLE DB, завдяки якій Microsoft Access може бути використаний для розробки клієнтських програм, що взаємодіють з іншими СУБД тощо.

Починаючи з MS Access 2007 використовується покращена версія ядра Microsoft Jet - **Microsoft Data Engine (MSDE).** Це обновлене ядро забезпечує інтеграцію з Windows SharePoint Services 3.0 і Microsoft Office Outlook 2007, а також нові можливості (наприклад, створення полів підстановок, які одночасно допускають кілька значень).

4. Проектування бази даних

Перш, ніж почати будувати свою базу даних, її необхідно спроектувати, тобто визначити для збереження яких саме даних вона призначена, хто буде нею користуватись тощо. Сам **процес побудови** бази даних має свій життєвий цикл, який проходить **ряд етапів**:

1. **Аналіз даних** включає в себе етапи формулювання і аналізу вимог.
2. **Проектування бази даних** передбачає безпосереднє створення об'єктів та зв'язків бази даних.
3. **Ввід в експлуатацію** включає:
   - перенесення бази даних на робочу машину;
   - написання технічної документації;
   - передача користувачу;
   - аналіз функціонування, підтримка та модифікація бази даних.

Етап аналізу даних являється самим складним і самим головним. Як правило, він здійснюється в тісній взаємодії з замовником. Його можна поділити ще на ряд **підетапів**:

1. **Концептуальне проектування**, що включає збір, аналіз і редагування вимог до даних. Для цього здійснюються наступні дії:
   - дослідження предметної області, вивчення її структури;
   - визначення користувачів майбутньої бази даних;
   - визначення вимог користувачів до бази даних, тобто що кожен з користувачів хоче отримати в результаті створення БД;
   - визначення інформаційних об'єктів, які повинні бути присутні в базі даних та зв'язків між ними.

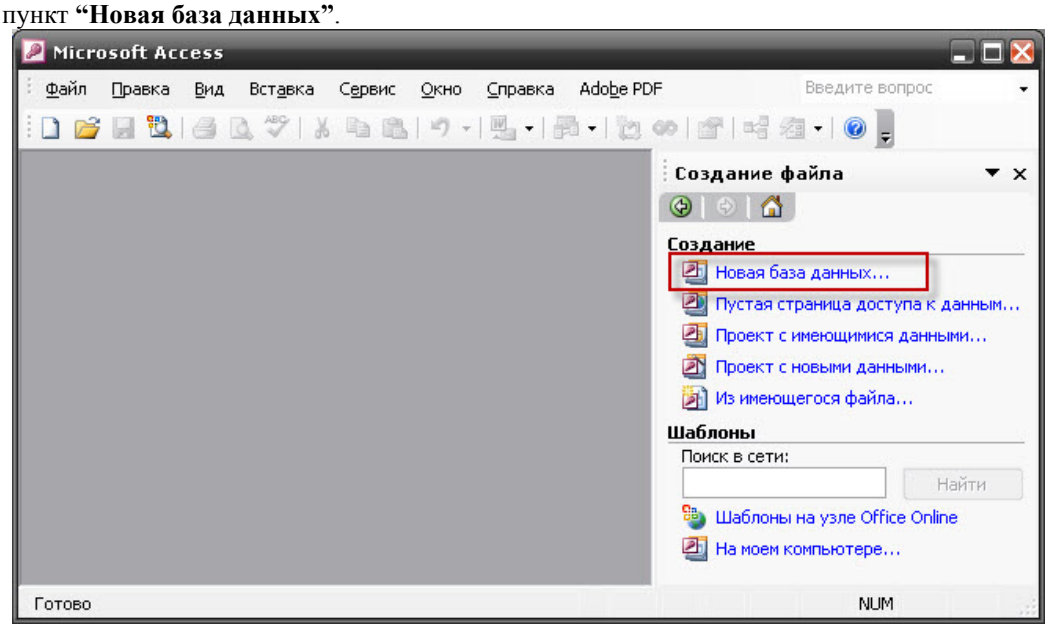
2. **Логічне проектування** – перетворення вимог до даних в конкретні об'єкти даних. В результаті, на виході ми отримуємо орієнтовну структуру бази даних.
3. **Фізичне проектування** – визначення особливостей зберігання даних (здійснюється групування даних, створюються індекси), методів доступу тощо.

Фактично, можна сказати, що концептуальний рівень проектування дозволяє представити базу даних з точки зору аналітика, логічний – з точки зору програміста, а фізичний – адміністратора.

5.Створення бази даних

Час перейти до практики. Після того, як Ви спроектували базу даних, її необхідно створити. Для демонстрації роботи з MS Access ми скористаємось версією, яка входить до складу пакета Microsoft Office 2003. Принцип створення бази даних в нижчих і старших версіях трішки відрізняється, але вцілому процес схожий. Тому, розібравшись з якоюсь однією версією, розібратись в іншій буде не досить складно.

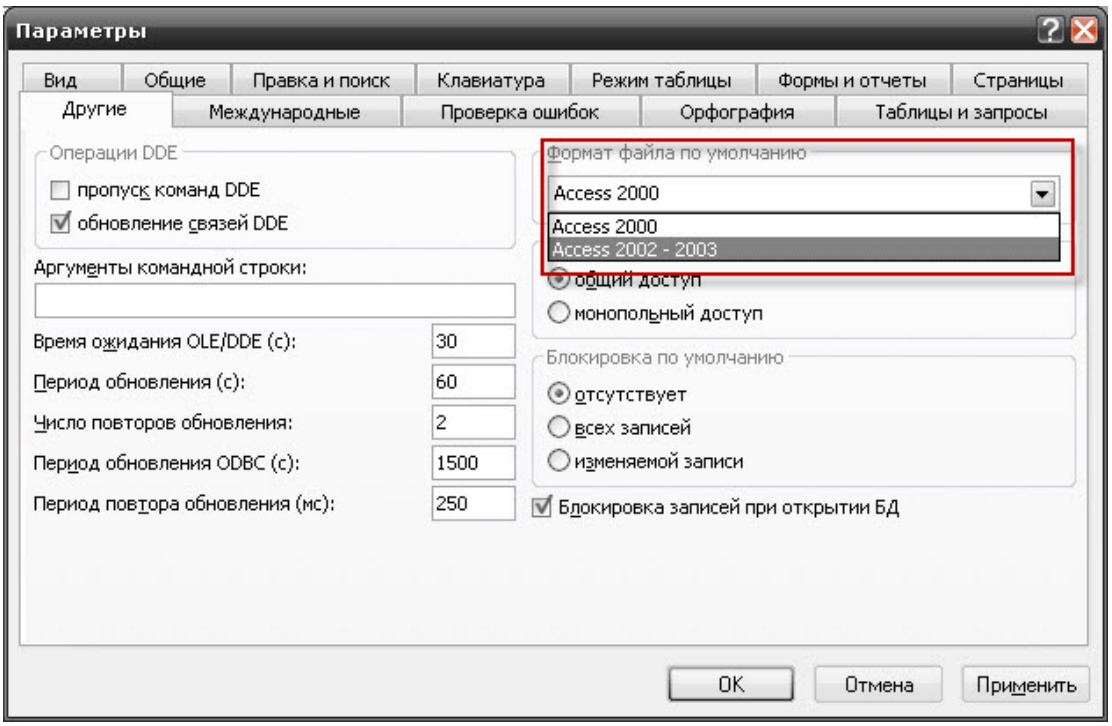
Отже, для того, щоб створити базу даних засобами MS Access потрібно перш за все його запустити. Після цього потрібно обрати пункт меню **Файл->Создать (Ctrl+N)**. Перед Вами з'явиться нова бокова панель з меню, в якому необхідно обрати пункт **"Новая база данных"**.



Це дозволить створити нову пусту базу даних, що буде зберігатися у файлі з ім'ям, яке Ви їй задасте на наступному кроці, з розширенням ***.mdb** та розміром близько 60 Kb.

Для відкриття вже існуючої бази даних слід скористатись пунктом меню **Файл->Открыть (Ctrl+O)**.

Після створення нової бази даних, Ви можете забезпечити її сумісність з базами даних інших (нижчих) версій СУБД MS Access. Для цього потрібно змінити прараметри новоствореної БД за допомогою пункта головного меню **Сервис->Параметры**. Потім обрати закладку «**Другие**» та в розділі «**Формат файла по умолчанию**» обрати необхідну версію.



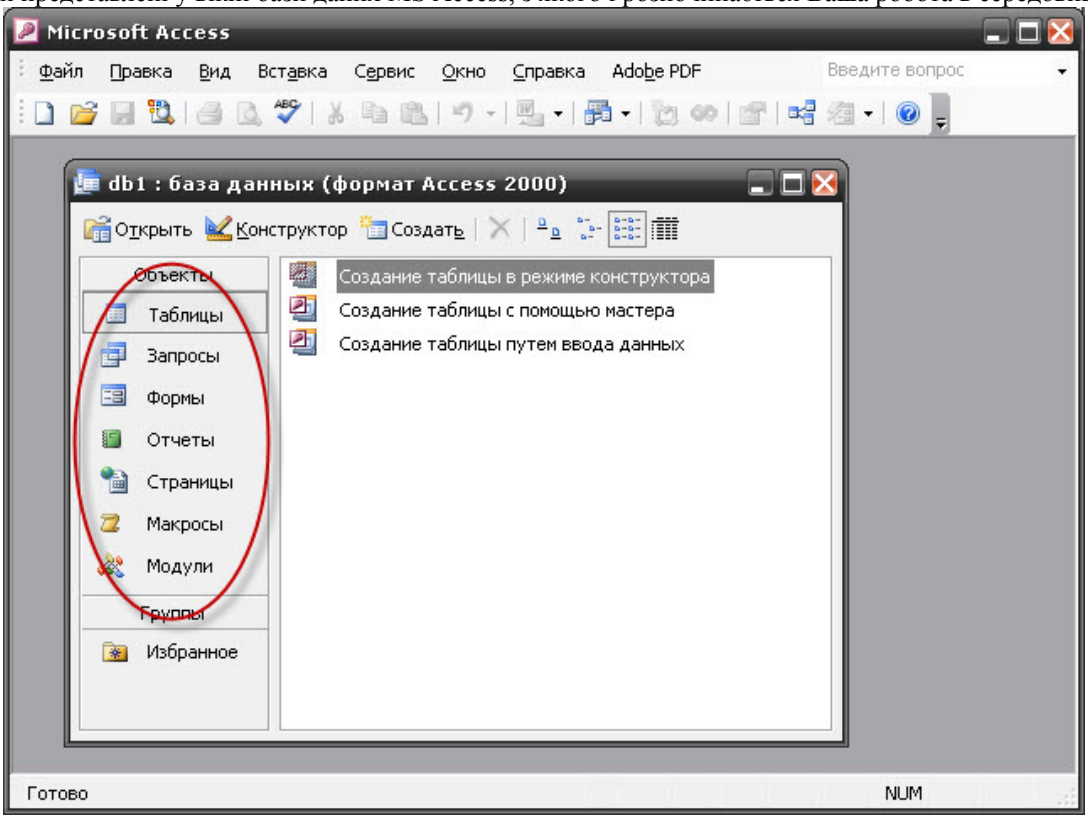
Слід відмітити, що дана можливість існує, починаючи з версії **MS Access 2002.**

Після того, як Ви визначились з вибором і проробили всі необхідні дії, перед Вами має постати нова база даних. З чим Вас і вітаємо (☺).

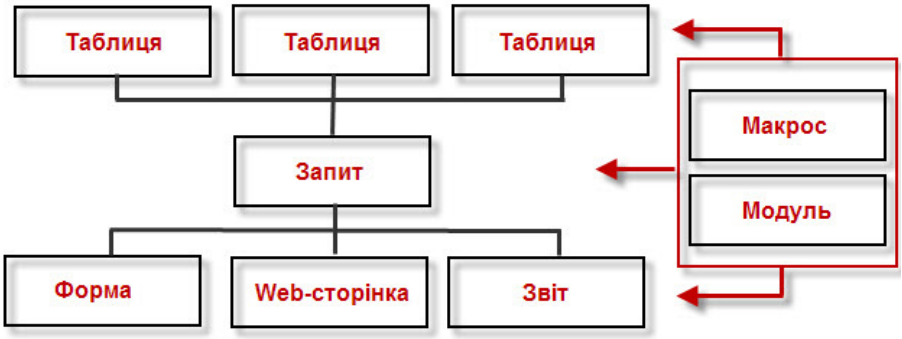
6.Складові частини бази даних MS Access

СУБД MS Access – це сукупність об’єктів різного типу і призначення, які використовуються для збереження, відображення і виводу на друк даних бази, а також можуть містити створений Вами програмний код. До об’єктів СУБД MS Access **відносяться**: таблиці, запити, форми, звіти, сторінки, макроси і модулі.

Всі об’єкти представлені у вікні бази даних MS Access, з якого і розпочинається Ваша робота в середовищі.



Таблиці являються місцем зберігання данних, всі інші об’єти - це інструменти, які призначені для роботи з таблицями і для створення користувацького інтерфейсу. Схематично взаємодію об’єктів СУБД можна зобразити наступним чином:



Таблиці – це об'єкти, які створює користувач для зберігання інформації, тобто свої дані. Access дозволяє створювати та редактувати таблиці. Таблиці складаються з рядків (записів, кортежів) та стовпчиків (полів, атрибутів). В свою чергу кожен стовпчик має свій формат. Одразу відмітимо, що таблиці можна імпортувати або робити посилання на таблиці, розміщені в інших СУБД.

Запити створюються користувачем для вибірки необхідних даних з однієї або декількох зв'язаних таблиць. Результатом виконання запитів є таблиця, яка може бути використана разом з іншими таблицями БД при обробці даних. В MS Access запит може формуватися у варіанті **Query By Example** (запит по зразку) або за допомогою мови структурованих запитів **Structured Query Language**. Відмітимо, що за допомогою запитів можна також управляти побудовою БД, наприклад, створювати нові таблиці, на основі вже існуючих.

Форми. Дозволяють створити інтерфейс, за допомогою якого користувачі можуть працювати з таблицями в більш зручній формі. Користувацбкий інтерфейс можна створювати як на основі таблиць, так і на основі запитів (в основному).

Звіти – це об'єкт, який використовується для виводу даних на друк в зручному вигляді. Варто відмітити, що звіти будуються переважно на основі запитів.

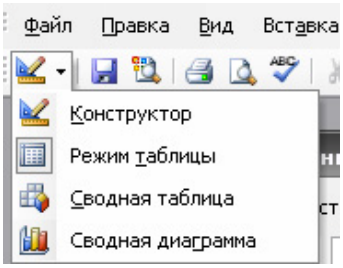
Сторінки – це HTML-сторінки, які дозволяють представити дані у вікні браузера, створити Web-інтерфейс доступу до даних. Це дозволяє звертатись до БД через Internet.

Макроси – це прості програми, які дозволяють за допомогою готових команд автоматизувати виконання часто повторюваних операцій. Кожна дія реалізується макрокомандою.

Модулі – це об'єкти БД, що містять програмний код, написаний на мові VBA (Visual Basic for Application). Як правило, вони використовуються як заміна макросам, або ж для написання власного коду реакції на події.

7. Режими роботи MS Access

MS Access дозволяє працювати з кожним об'єктом бази даних в різних режимах. Варто відмітити, що в основному ці режими індивідуальні для кожного об'єкта, але є ряд спільних. Режим роботи можна змінити за допомогою кнопки на панелі інструментів **«Вид»** в будь-який зручний для Вас момент часу. Розглянемо коротко існуючі режими роботи.



Таблиці бази даних мають 4 режими роботи:

1. **Режим конструктора**, в якому Ви можете змінювати структуру таблиці та її зовнішній вигляд. Цей режим призначений для програміста баз даних і в користувацьких додатках його потрібно блокувати.
2. **Режим таблиці**. Дозволяє переглянути дані в звичному для користувача режимі таблиці.
3. **Режими підсумкової таблиці і підсумкової діаграми**. Передбачає перегляд даних у відповідному вигляді.

Об'єкт бази даних **«Запити»** має додатковий режим роботи – **Режим SQL**, який дозволяє перейти у вікно для побудови запитів на мові SQL. З цим режимом Ви познайомитесь через кілька пар більш детальніше.

Об'єкт **«Форми»** має режими роботи, аналогічні об'єкту «Таблиці», і включає додатковий **Режим форми**. В цьому режимі Ви можете переглянути який вигляд буде мати форма для користувача. Режим конструктора для форми дозволять з легкістю розмістити необхідні елементи управління, через які буде відбуватись взаємодія користувача з даними бази даних.

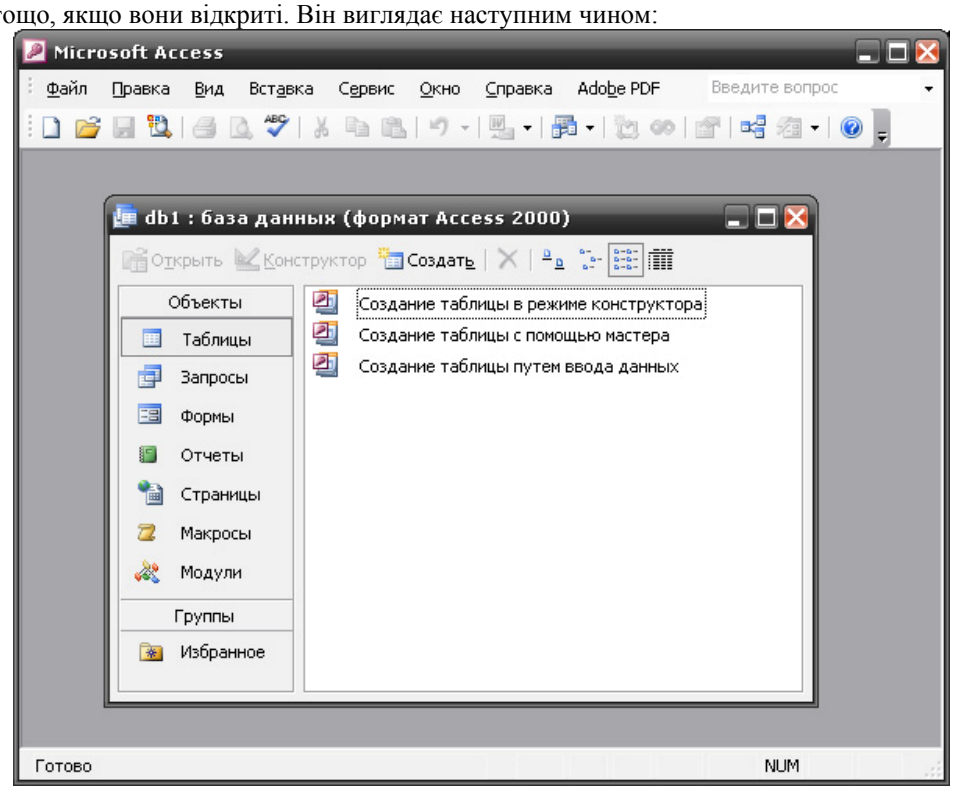
Звіти мають 3 режими роботи:

1. **Режим конструктора** – аналогічний попереднім.
2. **Режим попереднього перегляду** (Print Preview) дозволяє переглянути звіт в такому вигляді, який він буде мати при друці.
3. **Режим перегляду зразка** (Layout Preview) дуже зручний для перевірки структури і макета звіту. В цьому режимі на екран виводяться лише ті дані, які необхідні для заповнення кожного з елементів звіту.

Сторінки мають також 3 режими:

1. **Режим конструктора**.
2. **Режим перегляду сторінки**, який дозволяє впереглянути зовнішній вигляд сторінки в вікні бази даних.
3. **Режим попереднього перегляду веб-сторінки**, що запускає сторінку в браузері для перегляду.

Варто відмітити, що в MS Access виділяють ще один режим роботи – **початковий**. Цей режим роботи стосується бази даних вцілому. Він встановлюється на самому початку роботи і повернутися до нього можна лише закривши всі таблиці, форми, звіти, запити тощо, якщо вони відкриті. Він виглядає наступним чином:



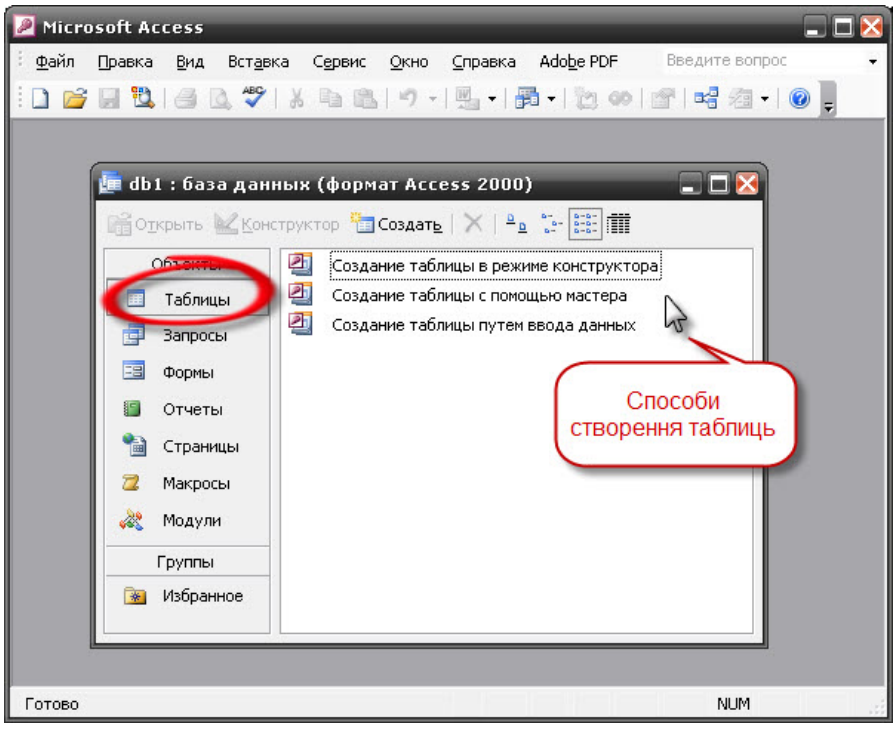
8. Таблиці. Порядок та методи створення таблиць в MS Access

Створивши базу даних, потрібно заповнити її даними. Дані, як вже було не раз сказано, зберігаються в таблицях, які зв'язані між собою зв'язками. Всі дані повинні бути організовані в таблицях таким чином, щоб забезпечити об'єднання різнорідної інформації, виключити дублювання даних та забезпечити швидкий і ефективний доступ до даних. Важливою особливістю таблиць є те, що представлена в них інформація описує, як правило, однотипні предмети або об'єкти. По суті, це список об'єктів, які мають спільні характеристики. Характеристики об'єктів вказуються в полях таблиці, а їх значення в відповідних комірках рядка таблиці, утворюючи запис.

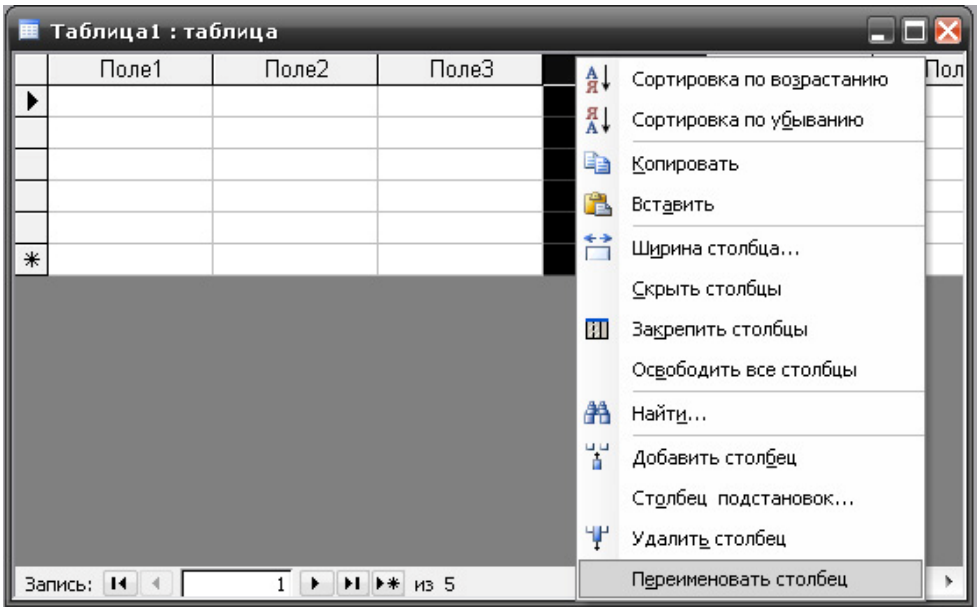
СУБД MS Access дозволяє створювати таблиці одним з **трьох способів**:

1. в режимі конструктора;
2. за допомогою візарда, тобто на основі шаблону;
3. шляхом введення даних в таблицю.

Обрати необхідний спосіб можна за допомогою необхідного ярлика в вікні активної бази даних, поряд з списком створених об'єктів, або обравши пункт меню «Создать» на панелі інструментів цього ж вікна та обравши необхідний список в списку.

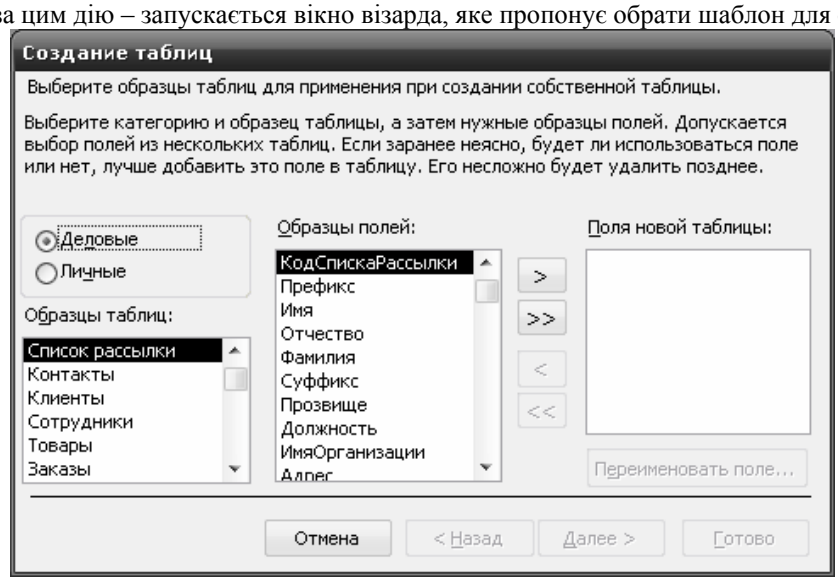


Розглянемо всі три методи і почнемо з самого простого способу створення таблиць – **шляхом введення даних**. При виборі даного режиму перед Вами з'явиться пуста таблиця, що має вигляд звичайної електронної таблиці MS Excel. Полям даної таблиці за допомогою контекстного меню можна надати необхідні назви характеристик, а в записи таблиці ввести необхідні дані.



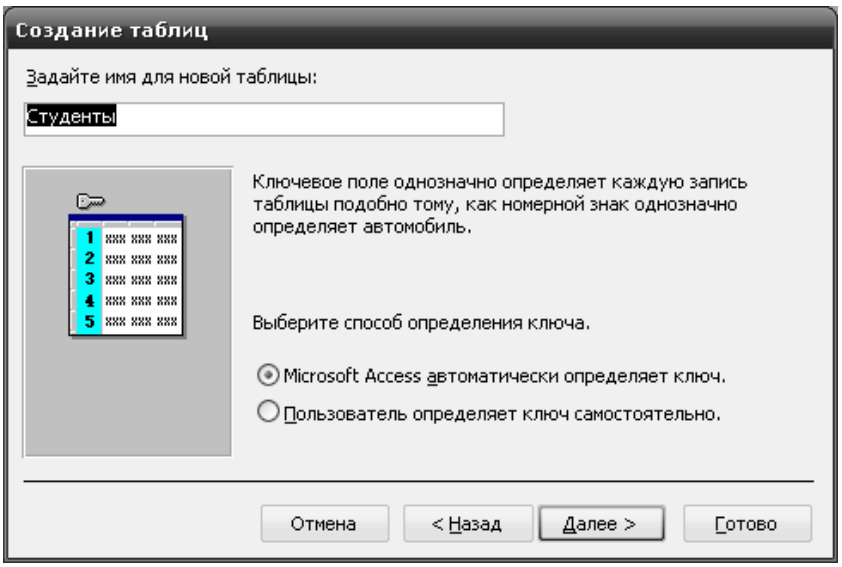
Практична частина. Щоб Ви більш детальніше уявляли собі даний процес створення таблиці цим способом створіть таблицю з назвою Student, що містить інформацію про студента та має наступну структуру: name, surname, dateOfBirth, course.

Наступний спосіб – **створення таблиці на основі шаблону**. Обираємо необхідний нам метод створення і спостерігаємо наступну за цим дію – запускається вікно візарда, яке пропонує обрати шаблон для створення таблиці.

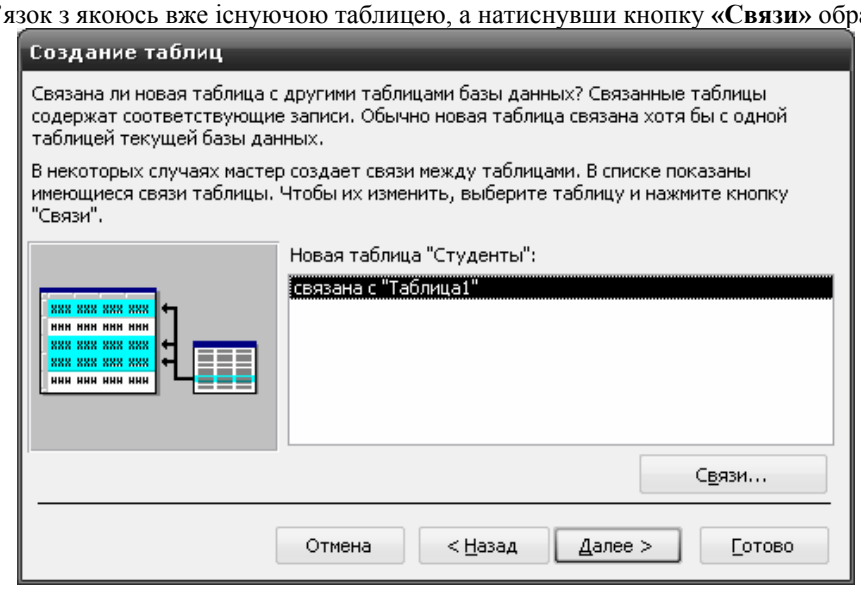


В полі **"Образцы таблицы"** обираємо шаблон майбутньої таблиці. В полі **"Образцы полей"** обираємо необхідні поля та переносимо їх за допомогою стрілок в поле з назвою **"Поля новой таблицы"**. Доречі, назви обраних полів можна змінити шляхом вибору кнопки **"Переименовать поле"**.

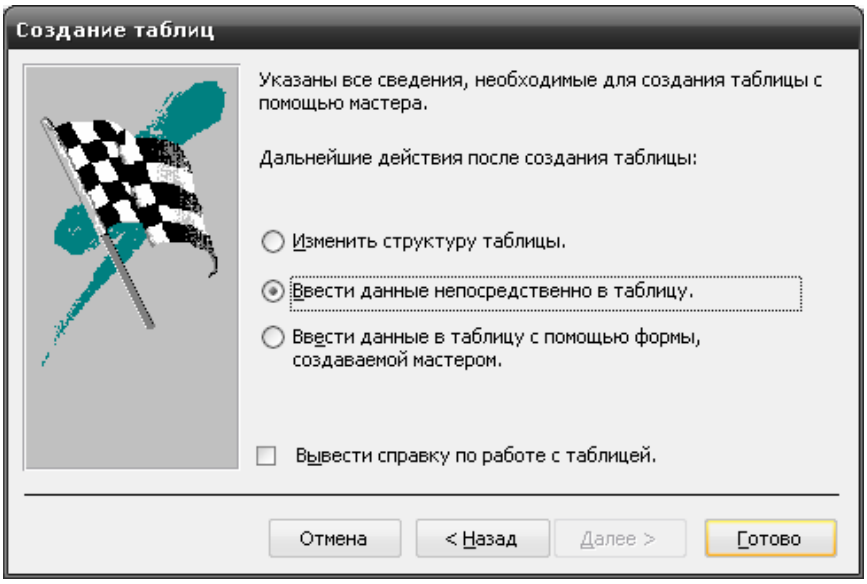
На наступному кроці можна задати ім’я новоствореної таблиці, якщо Ви не згідні з іменем, яке пропонує візард, а також первинний ключ. Про ключі ми поговоримо з Вами дещо пізніше, тому залишаємо дану закладку без змін і йдемо далі.



Далі можна задати зв’язок з якоюсь вже існуючою таблицею, а натиснувши кнопку **«Связи»** обрати тип зв’язку.



Останнім кроком необхідно вказати дію, що буде мати місце після створення даної таблиці. Обираємо необхідну та завершуємо роботу візарда.



Практична частина. Спробуйте створити будь-яку таблицю на основі запропонованих шаблонів, попереіменовувати деякі поля тощо.

Крім розглянутих вище двох способів створення таблиці БД, існує третій – **за допомогою конструктора**. Він надає повний контроль над полями створюваної таблиці, простий та зручний в роботі. Оскільки даний спосіб найчастіше використовують програмісти баз даних, то саме його ми і розглянемо більш детальніше та будемо використовувати в нашій подільшій роботі при додаванні таблиць до бази даних.

Отже, обираємо об'єкт "**Таблицы"** і варіант "**Создание таблицы в режиме конструктора"**, або ж обираємо кнопку на панелі інструментів "**Создать"** та варіант "**Конструктор"**. Тобто виконуємо стандартний набір дій по створенню таблиці.

Після цього перед Вами з'явиться вікно конструктора таблиці. Проаналізуємо його вміст.

В верхній частині вікна є три стовпчики: **Имя поля** (назва поля таблиці), **Тип данных** (тип даних, з якими ми будемо працювати: текст, числа тощо) і **Описание** (короткий коментар до даного поля).

В нижній частині вікна, представлені властивості даного поля у вигляді двох закладок: **Общие** (дозволяє налаштувати дане поле. Наприклад: для текстового типу даних можна вказати його розмір, для типу Дата/время - формат роботи з датою тощо) і **Подстановка** (дозволяє асоціювати дане поле з деяким елементом управління, який буде використаний при створенні форми таблиці).

Щоб краще зрозуміти основний принцип побудови таблиці в режимі конструктора потрібно для початку більш детальніше познайомитись з поняттям типу полів, тож перейдемо до цього знайомства.

9. Типи полів

СУБД MS Access підтримує 10 типів полів, короткі характеристики яких представлені нижче.

Текстовий (Text) – тип даних по замовчуванню. Текст або числа, що не приймають участі в розрахунках. Кількість символів в полі не повинно перевищувати 255. Максимальна кількість символів, яку можна ввести в поле, задається в властивості **Размер поля**. Пусті символи в невикористовуваних полях не зберігаються.

Поле MEMO (MEMO) – довгий текст, наприклад, опис певного об'єкта або примітка. Максимальна довжина 64 000 символів.

Числовий (Number) – числові дані, що використовуються в математичних розрахунках. Конкретні варіанти числового типу і їх довжина задаються в властивості **Размер поля**.

Грошовий (Date/Time) – грошові значення та числові дані, що використовуються в математичних розрахунках. Точність - до 12 знаків в цілій та до 4 знаків в десятковій частині. Довжина поля 8 байт.

Дата/Час (Currency) – значення дати або часу, що відносяться до років з 100 по 9999 включно. Довжина поля 8 байт.

Лічильник (Счетчик, AutoNumber) - тип даних поля, в яке для кожного нового запису автоматично вводяться унікальні цілі числа в діапазоні від 1 до двох з лишнім міліардів. Значення цього типу неможливо змінити чи видалити. Використовується для визначення унікального ключа таблиці, тому даний тип може мати лише одне єдине поле таблиці.

Логічний (Yes/No) – логічні дані, тобто поле, що може містити одне з двох можливих значень: Да/Нет; Истина/Ложь; Вкл./Выкл. тощо

Поле об'єкта OLE (OLE object) – об'єкт (наприклад, малюнок, звукозапис, електронна таблиця Microsoft Excel тощо), що зв'язаний або вбудований в таблицю MS Access. Довжина поля - до 1 Гб. Для полів типу OLE і MEMO не допускається сортування і індексування.

Гіперпосилання (Hyperlink) - в якості гіперпосилання можна вказувати шлях до файла на диску або URL-адресу. Максимальна довжина - 64000 символи.

Майстер підстановок (Lookup Wizard) – створює поле, в якому пропонується вибір значень з списку, або ж з поля із списком, що містить набір постійних значень або значень з іншої таблиці. Вибір даного типу поля запускає візард, що будує для поля список значень на основі поля або полів з іншої таблиці.

10. Домашнє завдання

Створити базу даних підприємства, яка складається з трьох таблиць, що містять інформацію про:

- Працівників;
- Відділи;
- Продукцію.

Продумати структуру кожної таблиці та створити їх різними способами.