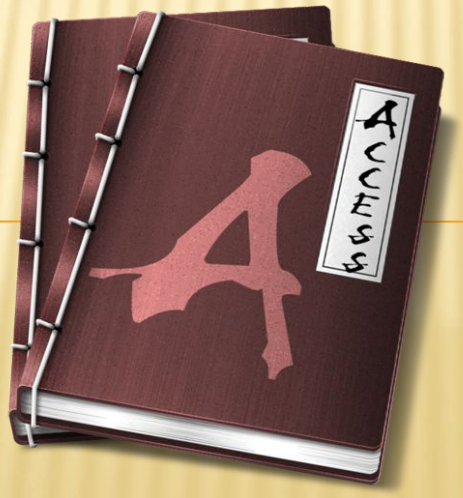
Інформатика 11 клас

Урок

Проектування бази даних. Основні елементи реляційної бази даних.



Як побудувати модель даних предметної області?

Процес побудови моделі даних предметної області належить до абстрактного моделювання і, як правило, поділяється на кілька етапів, що базуються на системному підході. Ці етапи розглянемо на прикладі. Побудуємо модель бази даних *Бібліотека.*

1. Визначення мети створення бази даних.

На першому етапі побудови моделі необхідно визначити мету створення бази даних, основні її функції і набір даних, тобто визначити основний зміст таблиць бази даних і дані, які будуть зберігатися в полях таблиць.

Під час роботи з базою даних наприклад ***Бібліотека*** користувач може шукати відповіді на такі запитання:

1. Чи є у бібліотеці конкретна книга?
2. Чи є книга в наявності?
3. Книги яких авторів даної тематики є у бібліотеці?
4. Які книги даної тематики видано в заданий період?
5. Які видавництва публікують книги даної тематики?
6. Яке видавництво видало дану книгу?
7. Як зв'язатися з видавництвом для того, щоб замовити книги?

Таким чином, можна так сформулювати мету створення цієї бази даних: зберігати дані про книги, авторів та видавництва для подальшого задоволення запитів користувачів.

Як побудувати модель даних предметної області?

2. Розробка таблиць, з яких складатиметься база даних.

Одним з найбільш складних етапів у процесі побудови моделі є розробка таблиць, оскільки очікувані результати не завжди дають повне уявлення про структуру таблиць.

Наприклад, відносно бази даних *Бібліотека* можна виділити три об'єкти (сутності), характеристики яких можна зберігати в таблицях, а саме: книга, автор,

Книга	*Автор*	*Видавництво*
назва книги, автор, кількість сторінок, видавництво, рік видання, ціна, наявність у сховищі, тематика, ключові слова;	прізвище, ім'я, по батькові, жанр творчості;	назва, адреса, спеціалізація, сайт.

Для опису кожного виділеного об'єкта доцільно побудувати таблицю, до якої внести імена атрибутів об'єктів та типи значень цих атрибутів, а також обмеження на їх значення.

Як побудувати модель даних предметної області?

Проектування бази даних передбачає розробку структури таблиць та встановлення зв'язків між ними. Внесення даних та їх опрацювання не стосується етапу проектування бази даних.

Під час проектування таблиць спочатку краще розробити їх структуру на папері, при цьому доцільно користуватися основними правилами:

Основні правила

Дані в таблиці не повинні дублюватися, також не має бути повторень між таблицями. Якщо деякі дані зберігатимуться тільки в одній таблиці, то і змінювати їх доведеться тільки в одному місці. Наприклад, у таблиці *Книга* може міститися назва книги, її автор, видавництво, рік видання. Ці відомості не повинні повторюватися в інших таблицях, достатньо ввести поле *Книга,* для якого встановити зв'язок із ключовим полем таблиці *Книга* (у даному прикладі це може бути поле *Код_книги,* яке слід створити додатково).

Основні правила

Кожна таблиця має містити дані лише з однієї теми. Дані з кожної теми опрацьовуються значно легше, якщо вони містяться в незалежних одна від одної таблицях. Наприклад, адреса видавництва і відомості про книги зберігаються в різних таблицях для того, щоб при видаленні відомостей про книгу дані стосовно видавництва залишилися у базі даних, а при закритті видавництва відомості про книги, що ви дані ним раніше, не були також видалені.

Як побудувати модель даних предметної області?

3. Визначення полів таблиці.

Кожна таблиця містить дані окремої теми, а кожне поле в таблиці містить окреме значення. Наприклад, у таблиці з даними про авторів можуть міститися поля з прізвищем, адресою і номером телефону. Під час розробки полів для кожної таблиці необхідно пам'ятати:

- кожне поле має відповідати темі таблиці;
- не рекомендується включати до таблиці дані, які є результатом обчислення значень виразів;
- в таблиці слід розміщувати всі необхідні дані;
- дані бажано поділяти на найменші логічні одиниці (наприклад, поля *Ім'я* і *Прізвище,* а не загальне поле *ПІП,* поля *Місто, Вулиця, Будинок, Квартира,* а не загальне поле *Адреса).*

Як побудувати модель даних предметної області?

4**. Визначення ключа таблиці.**

Кожна таблиця має містити поле або набір полів, що будуть задавати унікальне значення кожного запису в таблиці, за яким можна точно визначити потрібний запис. Таке поле або набір полів називають ***основним ключем***.

5. Визначення зв'язків між таблицями.

Після розподілу даних у таблицях і визначення ключових полів необхідно вибрати схему для зв'язування даних у різних таблицях. Для цього слід визначити зв'язки між таблицями. Наприклад, *Автор - Книга* (зв'язок 1-N), *Видавництво - Книга* (зв'язок 1-N) тощо.

6. Оновлення структури бази даних.

Після опису таблиць, полів і зв'язків необхідно ще раз переглянути структуру бази даних і виявити можливі недоліки. Крім того, необхідно виключити з таблиць усі можливі повторення даних.

У процесі побудови моделі важливо враховувати основні вимоги — модель має бути універсальною, адекватною, точною та економічною щодо визначеної предметної області.

За допомогою програми Access можна:

- додавати нові дані до бази даних, наприклад новий предмет до інвентарного списку;
- редагувати наявні дані бази даних, наприклад змінювати поточне розташування предмета;
- видаляти відомості, якщо, наприклад, предмет було продано або списано;
- впорядковувати та переглядати дані різними способами;
- спільно користуватися даними з іншими користувачами за допомогою звітів, повідомлень електронної пошти, інтрамережі або Інтернету

Основні функції СУБД ACCESS

1. Проектування базових об'єктів – двовимірні таблиці з полями, що містять дані різних типів;
2. Створення зв'язків між таблицями з підтримкою цілісності даних, каскадного оновлення полів та каскадного видалення записів;
3. Введення, збереження, перегляд, впорядкування, зміна та вибір даних з таблиць із використанням різних засобів контролю даних, індексування таблиць та засобів алгебри логіки;
4. Створення, модифікація та використання похідних об'єктів (запитів, форм, звітів).

ОСНОВНІ ОБ’ЄКТИ **СУБД ACCESS**

Таблиці

Таблиця бази даних схожа на електронну таблицю, в якій дані зберігаються в рядках і стовпцях. В результаті зазвичай досить легко імпортувати електронну таблицю до таблиці бази даних. Головна відмінність між збереженням даних в електронній таблиці та базі даних — це спосіб упорядкування даних.

Щоб забезпечити максимальну гнучкість бази даних, дані необхідно впорядкувати в таблицях, щоб позбутися зайвих елементів. Наприклад, якщо потрібно зберігати дані про працівників, відомості про кожного працівника необхідно один раз ввести в таблиці, яка настроєна лише для розміщення даних про працівників. Цей процес називається **оптимізацією.**

Кожний рядок у таблиці називається записом. Записи — це місце розташування окремих елементів даних.

Кожний запис складається з одного або кількох полів. Поля відповідають стовпцям у таблиці. Наприклад, можна створити таблицю «Працівники», де кожний запис (рядок) зберігає відомості про окремого працівника, а кожне поле (стовпець) містить власний тип даних, наприклад ім'я, прізвище, адресу тощо. Поля мають містити певний тип даних: текст, дату або час, число або інший тип.

Ще один спосіб опису записів і полів: уявіть старий картковий каталог у бібліотеці. Кожна картка у ящику відповідає запису бази даних. Кожний елемент даних на окремій картці (автор, назва тощо) відповідає полю бази даних.

Форми

Форми іноді називаються «екранами вводу даних». Це інтерфейси, які використовуються під час роботи з даними, тому вони часто містять кнопки для виконання різних команд. Можна створити базу даних без використання форм, просто редагуючи дані в таблицях даних. Проте більшість користувачів баз даних використовують форми для перегляду, введення та редагування даних у таблицях.

Форми пропонують простий у використанні формат роботи з даними, крім того, до них можна також додавати функціональні елементи, наприклад кнопки. Ці кнопки можна настроїти для визначення даних, що відображатимуться у формі, відкриття інших форм або звітів та для виконання низки інших завдань. Наприклад, є форма «Форма клієнта», у якій виконується робота з даними клієнта. Форма клієнта може містити кнопку, яка відкриває форму замовлення, де можна ввести нове замовлення цього клієнта.

Форми також дають змогу керувати способом взаємодії інших користувачів із даними бази даних. Наприклад, можна створити форму, яка відображає лише певні поля та дозволяє виконувати лише певні операції. Це допомагає захистити дані та забезпечує належне введення даних.

Звіти

Звіти використовуються для зведення та представлення даних у таблицях. Звіт зазвичай відповідає на певне питання, наприклад «Яку суму було отримано від кожного клієнта цього року?» або «У яких містах розташовані наші клієнти?». кожний звіт можна відформатувати таким чином, щоб він представляв дані найбільш зрозумілим способом.

Звіт можна запустити будь-коли, і він завжди відображатиме поточні дані в базі даних. Звіти зазвичай мають формат для друку, але їх також можна переглядати на екрані, експортувати до іншої програми або надсилати електронною поштою.

Запити

Запити — це справжні робочі коники бази даних, які можуть виконувати багато різних функцій. Їх найпоширеніша функція — отримання певних даних із таблиць. Дані, які потрібно переглянути, зазвичай розташовані в кількох таблицях, і запити дають змогу переглянути їх в одній таблиці даних. Також, оскільки зазвичай не потрібно бачити всі записи одночасно, запити дозволяють додавати критерії для «фільтрування» даних, щоб переглядати лише потрібні записи. Запити часто виконують роль джерела записів для форм і звітів.

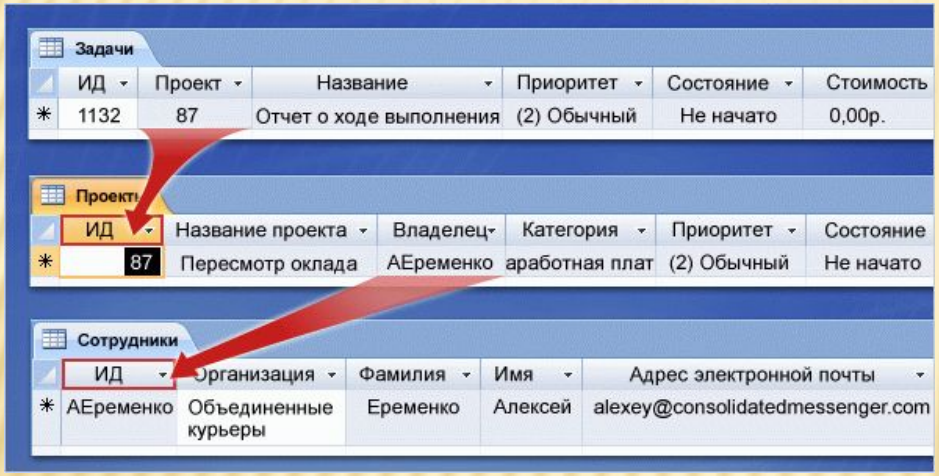
Певні запити є «оновлюваними», тобто дані в базових таблицях можна редагувати за допомогою таблиці даних запиту. Якщо дії виконуються з оновлюваним запитом, слід пам'ятати, що зміни насправді виконуються в таблицях, а не лише в таблиці даних запиту.

Запити поділяються на дві основні групи: запити на вибірку і запити на дію.

Запит на вибірку просто отримує дані й робить їх доступними для використання. Результати запиту можна переглянути на екрані, роздрукувати або скопіювати до буфера обміну. Або можна використати результат запиту як джерело записів для форми чи звіту.

Запит на змінення, згідно з назвою, виконує з даними певне завдання. Запити на змінення можна використовувати для створення нових таблиць, додавання даних до наявних таблиць, оновлення або видалення даних.

Дані можна ефективно використовувати завдяки пов'язаних таблиць



Access відрізняється від інших програм тим, що в ньому використовуються зв'язані таблиці. Він розроблений так, що одна таблиця може знаходити і використовувати дані з іншої таблиці. Завдяки цьому, наприклад, для відстеження поточних завдань не потрібно вводити вже збережені імена співробітників і назви проектів.

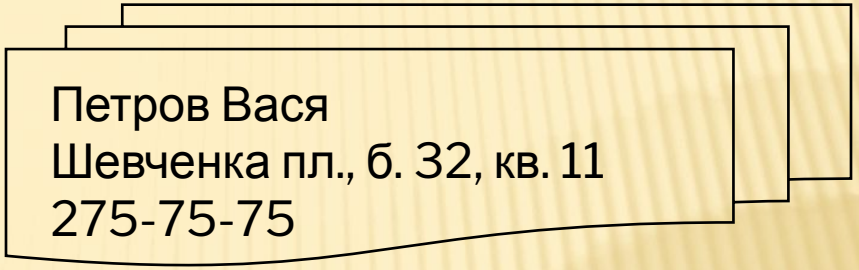
Це показано на малюнку. У таблицю "Завдання" вводиться нове завдання. Для цього, крім іншого, потрібно вибрати номер проекту в поле Проект, яке використовує дані з таблиці "Проекти". Крім того, поле Відповідальний в таблиці "Проекти" використовує дані з таблиці "Співробітники", а отже, і в таблиці "Завдання", і в таблиці "Проекти" є відомості про те, якому співробітникові призначена задача. Це дозволяє зрозуміти, кому призначено проект. Пов'язані таблиці роблять Access більш ефективним засобом, ніж списки або листи.

Реляційні БД

Модель – картотека

Приклади:

- записна книжка
- каталог в бібліотеці



поля

записи

Прізвище	Ім'я	Адреса	Телефон
Петров	Вася	Шевченка пл., б. 32, кв. 11	275-75-75
Іванов	Петро	Франка вул., б. 25, кв.12	276-76-76

1) найпростіша структура
2) всі інші типи БД використовують таблиці

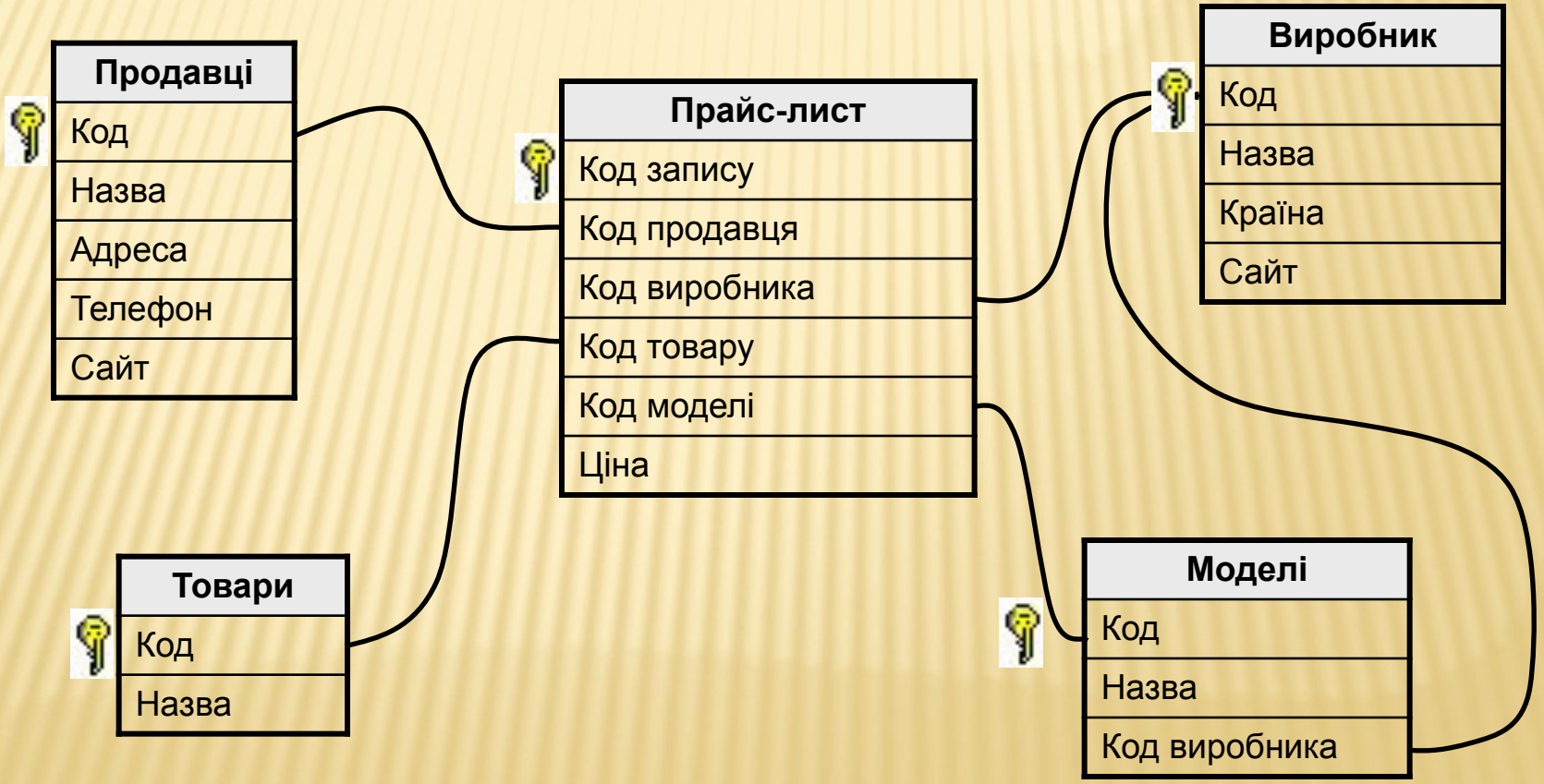
в багатьох випадках – дублювання даних:

Т.Г. Шевченко	Кобзар	540 ст.
Т.Г. Шевченко	Гайдамаки	45 ст.

Реляційні БД

1970-і рр. Е. Кодд, англ. *relation* – відношення.

Реляційна база даних – це набір простих таблиць, між якими встановлені зв'язки (відношення) з допомогою числових кодів.



Реляційні БД

1) немає дублювання інформації;
2) при зміні адреси фірми, достатньо змінити її тільки в таблиці Продавці;
3) захист від неправильного введення: можна вибирати тільки фірму, яка заздалегідь введена в таблицю Продавці;
4) механізм транзакції: будь-які зміни вносяться в базу тільки тоді, коли вони повністю завершені.

1) складність структури (не більше 40-50 таблиць);
2) при пошуку потрібно звертатися до декількох таблиць;
3) потрібно підтримувати цілісність: при вилученні фірми продавця потрібно вилучати всі зв'язані записи з всіх таблиць (в СКБД – автоматично, каскадні вилучення).

Зв'язки між таблицями

Один до одного («1-1») – одному запису в першій таблиці відповідає тільки один запис в другій таблиці.

Примітка: виділення часто використовуваних даних.

1 — 1

Код	Прізвище	Ім'я
1	Іванов	Кузьма
2	Петров	Василь
…		

Код	Рік народження	Адреса
1	1992	Шевченка, б.20, кв. 6
2	1993	Франка вул., б. 30, кв. 18
…		

Один до багатьох («1- ∞») – одному запису в першій таблиці відповідає декілька записів в другій.

товари

1 — ∞

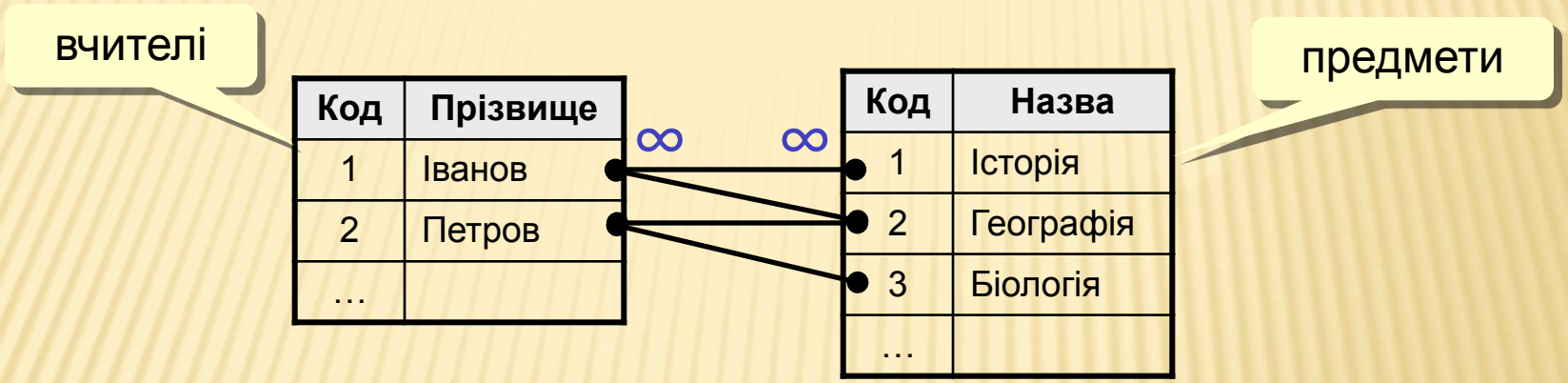
прайс-лист

Код	Назва
1	Монітор
2	Вінчестер
…	

Код	Код товару	Ціна
123	1	2 999
345	1	2 499
…		

Зв’язок між таблицями

Багато до багатьох («∞ - ∞») – одному запису в першій таблиці відповідає декілька записів в другій, і навпаки.



Код	Прізвище
1	Іванов
2	Петров
…	

Код	Назва
1	Історія
2	Географія
3	Біологія
…	

Реалізація – через третю таблицю і два зв'язки «1-∞».

Код	Прізвище
1	Іванов
2	Петров
…	

Код	Код вчителя	Код предмету	Клас
1	1	1	9-А
2	1	2	8-Б
3	2	3	7-В
…			

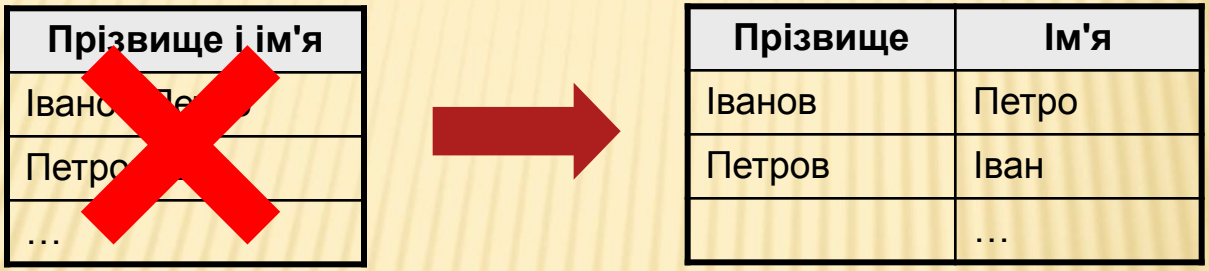
Код	Назва
1	Історія
2	Географія
3	Біологія
…	

Нормалізація бази даних

Нормалізація – це розробка такої структури БД, в які немає надлишкових даних і зв'язків.

Основні принципи:

- Будь-яке поле повинно бути неподільним.



- Не повинно бути полів, в яких позначені різні види одного і того ж, наприклад, товарів.

Нормалізація бази даних

Основні принципи:

- Будь-яке поле повинно залежати тільки від ключа (ключ – це поле або комбінація полів, однозначно визначає запис).

товари

Код	Назва	Ціна
1	Монітор	
2	Вінчестер	
…		

залежить не тільки від назви товару!

прайс-лист

- Не повинно бути полів, які можуть бути знайдені з допомогою інших.

Код	Товар	Ціна за тонну	Кількість, тонн	Вартість
1	Банани	240	10	
2	Киви	300	20	
…				