МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

ОРГАНІЗАЦІЯ БАЗ ДАНИХ ТА ЗНАНЬ

Курсова робота
для студентів спеціальності 122
“Комп’ютерні науки”

Київ 2020

Організація баз даних і знань
Курсова робота

Містить опис методів побудови реляційної моделі даних для довільної предметної області із подальшим створенням відповідної бази даних у середовищі системи управління базами даних. Розглянуто правила запису отриманих реляційних конструкцій і запитів до них на мові структурованих запитів SQL, а також правила вбудовування операторів SQL у програми на алгоритмічних мовах з метою маніпулювання базою даних.

Для студентів спеціальності 8.080401 "Інформаційні управляючі системи та технології".

ВСТУП

Курсова робота виконуються згідно з навчальною програмою дисципліни "Організація баз даних і знань", яка призначена для студентів спеціальності 122 « Комп'ютерні науки». Мета виконання Курсової роботи – набуття студентами практичних навиків та закріплення ними теоретичних знань з питань моделювання інформаційних об'єктів і процесів предметної області інформаційних систем, дослідження методів проектування баз даних та оволодіння методами прикладного програмування в середовищі конкретної системи управління базами даних (СУБД).

Загальною метою виконання курсової роботи є поглиблення та закріплення знань з розділів дисципліни "Організація баз даних і знань".

Для цього студент повинен

– одержати у викладача індивідуальний номер варіанта з описом предметної області інформаційної системи , для якої потрібно спроектувати базу даних;

– побудувати власну БД та виконати її адміністрування;

– розробити відповідні програми на мовах SQL та ESQL/C;

– зробити висновки щодо курсової роботи;

– підготувати звіт по виконаній курсовій роботі;

Звіт по курсовій роботі повинен містити:

1. Титульний аркуш.
2. Мету роботи.
3. Опис предметної області.
4. Стислі теоретичні відомості.
5. Порядок виконання роботи.
6. Діаграму «сутності-зв'язки»
7. Реляційну модель бази даних.
8. Прикладну програму ,яка виконує всі операції із спроектованою базою даних.
9. Основні висновки.

1. Короткі теоретичні відомості.

1.1. Побудова концептуальної моделі бази даних.

Застосований метод графічного моделювання базується на побудові E-R (Entity-Relationship) діаграми для подальшого перетворення її в реляційні конструкції. Основними етапами E-R інформаційного моделювання є:

1. Визначення основних інформаційних об'єктів та їх характеристик, до яких належать:
   - сутності;
   - зв'язки;
   - атрибути.
2. Побудова E-R діаграми інформаційних об'єктів.

Визначення основних інформаційних об'єктів та їх характеристик

Основними інформаційними об'єктами E-R моделі даних є сутності, а їх характеристиками – зв'язки й атрибути. Розглянемо процес визначення і дослідження об'єктів та їх характеристик.

Вибір і дослідження можливих сутностей

Сутності – це ті речі, котрі користувачі інформаційної системи вважають важливими в галузі, яка моделюється. Сутності подібні класам в об'єктно-орієнтованому проектуванні.

Першим кроком у моделюванні є вибір можливих сутностей. Виходячи з розуміння фундаментальних елементів БД, можна скласти попередній список сутностей і, обговоривши його з користувачем БД, залишити головні з них.

Після запису повного списку сутностей можна перейти до їх аналізу й скорочення списку. Після дослідження ми залишаємо в списку ті сутності, які задовольняють таким вимогам:

1. Сутність повинна бути значущою, тобто в списку залишаємо тільки важливі для користувача БД сутності.
2. Сутність повинна бути загальною. Це означає, що сутність повинна являти собою тип предмета, а не індивідуальні зразки.
3. Сутність повинна бути фундаментальною, тобто вона існує без необхідності її пояснювати, вона не може бути межею чогось,

особливістю або описом. У список не включаються ті, що можуть бути отримані з інших сутностей.

4. Сутність повинна бути неподільною. Кожна наведена в списку сутність повинна являти собою простий клас і не ділитися на підкатегорії, що мають властиві їм риси.

Розглянемо приклад. Припустимо, що ми створюємо БД для комп'ютеризації персонального телефонного довідника. У модель БД треба занести імена, адреси й телефонні номери людей і організацій, з якими користувач має справу в бізнесі й дозвіллі.

Визначимо сутності для моделі БД. Якщо подивитися на сторінку телефонного довідника, то побачимо, що вони там є. Виберемо як сутності:

- ім'я (особи або організації);
- адресу;
- телефонний номер.

Розглянемо, чи відповідають вони сформульованим вимогам.

Очевидно, що вони значущі й загальні для моделі БД. Для того, щоб перевірити фундаментальність сутностей, перевіримо, чи можуть вони змінюватися незалежно одна від одної. Оскільки деякі люди й організації можуть змінити адресу, не змінюючи номер телефону або навпаки, а один номер телефону або адреса може належати різним особам, робимо висновок про незалежність усіх трьох сутностей попарно однієї від іншої.

Перевіримо, чи є ці сутності неподільними. “Ім'я” може належати людині або організації. Для забезпечення неподільності цієї сутності необхідно забезпечити однакову форму адреси для осіб і організацій.

Проаналізувавши сутність “телефонний номер”, приходимо до висновку, що є три типи телефонних номерів:

– номер телефону для розмови людей;

– номер факсу, пов'язаного з факс-пристроєм;

– номер модема, пов'язаного з комп'ютером.

Після проведеного аналізу запишемо діаграму сутностей, яка для телефонного довідника має вигляд:

Визначення зв'язків

Після вибору сутностей необхідно визначити зв'язки між ними. Зв'язки не завжди очевидні. Тому необхідно на початку скласти список усіх можливих зв'язків, а потім шляхом їх аналізу виділити лише істотні для моделі БД.

Зв'язок є деякою асоціацією між сутностями й описується в термінах:

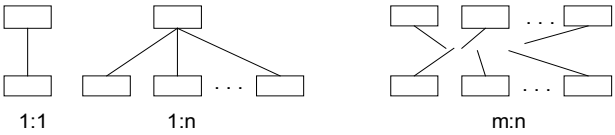
- зв’язаність;
- істотність залежності;
- кардинальність.

Зв’язаність належить до числа властивостей сутностей і описує кратність входження примірників деякої сутності в зв'язок.

Існує три типи зв’язаності:

- один до одного (1:1);
- один до багатьох (1:n);
- багато до багатьох (m:n).

Графічно види зв’язаності зображують таким чином:



Наприклад, у довіднику одна адреса може бути пов'язана більш ніж з одним ім'ям, тобто тип зв’язаності - 1:n.

Істотність залежності визначає, чи є дана сутність у зв'язку обов'язковою або необов'язковою. Для однозначного визначення цього факту необхідно сформулювати деякі бізнес-правила. Наприклад, можна зажадати, щоб ім'я було пов'язане з адресою, що робить істотну залежність зв'язку між сутностями “ім’я” і “адреса” обов'язковою. Прикладом необов'язкового зв'язку є правило, що особистість може мати або не мати дітей.

Кардинальність зв'язку накладає обмеження на те, скільки разів деяка сутність може бути наявна в зв'язку. Кардинальність зв'язку 1:1 також 1, але кардинальність зв'язку 1:n (n - деяке число)

невизначена (none). Звичайно встановлюється верхня межа для числа n , яка і вважається кардинальністю зв'язку.

Дослідження зв'язків

Одним зі шляхів дослідження зв'язків є побудова матриці зв'язків, яка має для телефонного довідника такий вигляд:

	name	address	phon	fax	modem
name	none	0-n 0-1	1 0-n	1-n 0-n	1 0-n
address	-	none	none	none	none
phon	-	-	none	none	none
fax	-	-	-	none	none
modem	-	-	-	-	none

Зв'язок сутності A з нею ж вважаємо невизначеною (none). Якщо зв'язок між якими-небудь іншими сутностями також не визначений, то у відповідній клітині також пишемо none.

Вважаємо, що ім'я може бути пов'язане з одним або з 0 адресами. Тому пишемо внизу відповідної клітини 0-1. Адреса, в свою чергу, може бути пов'язана з багатьма людьми, або з жодним. Тому у верхній частині клітини (name-address) пишемо 0-n. Тепер розглянемо зв'язок номера телефону з ім'ям (name-phon). Очевидно, що одна людина (організація) може мати багато номерів телефону (домашній, офісний, пейджер, мобільний тощо), але може не мати жодного. Тому в нижній частині клітини (name-phon) пишемо 0-n. Продовжуємо заповнювати клітинки матриці, керуючись такими бізнес-правилами:

- ім'я може бути пов'язане більш ніж з одним номером факсу. Наприклад, організація може мати декілька факсів. З іншого боку, номер факсу може бути пов'язаний більш ніж з одним ім'ям. Наприклад, різні співробітники організації можуть використати один і той самий факс-номер;

- номер модема повинен бути пов'язаний тільки з одним ім'ям (хоча це не очевидно). Однак ім'я може бути пов'язане більш ніж з одним модемом.

У таблиці також відображено таке міркування: не існує зв'язку між номером факсу і модема, між номером телефону і факсу, номером телефону і модема.

Ідентифікація атрибутів

Сутності мають властивості, які описуються атрибутами. Атрибут – це неподільна частина інформації про сутність, а для визначення атрибутів необхідно ідентифікувати сутності, тобто визначити, які характеристики необхідні для того, щоб все знати про дану сутність.

Виділені атрибути повинні мати такі властивості:

- значущість;
- прямі (тобто такі, що не виводяться з інших);
- неподільність;
- повинні містити дані одного типу.

На основі аналізу сутностей, враховуючи вимоги, складемо список атрибутів і занесемо їх у таблиці:

name	address	phon	fax	modem
fname	street	vce_num		mdm_num
			fax_num	
lname	city	vce_type	oper_from	b1300
bday	state		oper_till	b1200
anniv	zipcode			b2400
e-mail				
child1				
child2				
child3				

1.2.Побудова E-R діаграми інформаційних об'єктів

Отже, коли основні елементи інформаційної моделі визначені, можна подати їх у вигляді E-R діаграми (рис. 1).

Метод моделювання за допомогою E-R діаграм дає можливість вирішувати такі задачі:

- моделювати інформаційні потреби організації;
- визначати сутності і зв'язки;

- заздалегідь визначати дані;
- документувати процес обробки додатків;
- відображати логічну структуру БД для подальшого перетворення її в фізичну схему.

Розглянемо основні принципи побудови E-R діаграми на прикладі сутностей name і address, використовуючи нотацію Чена.

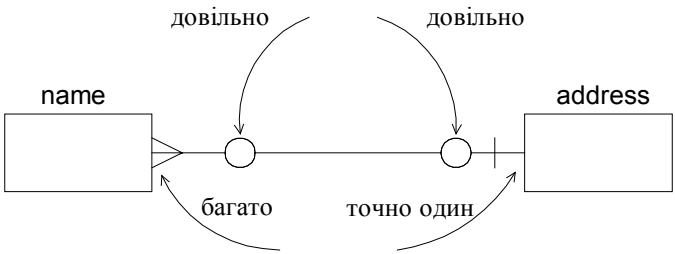


Рис. 1. Приклад E-R діаграми

Сутності на E-R діаграмі зображуються прямокутниками, а зв'язки – лініями, що з'єднують їх. Для зображення невизначеності зв'язку використовується коло, для зображення того, що зв'язок тільки з одним примірником – риска, а для зображення зв'язку з багатьма примірниками – трикутник. Діаграма читається ліворуч-праворуч, а потім праворуч-ліворуч. Наприклад, діаграма на рис. 1 читається так: “ім'я може бути пов'язане з 0 або точно з 1 адресою”, “адреса може бути пов'язана з 0 або багатьма іменами”.

Побудуємо E-R діаграму телефонного довідника (рис. 2), використовуючи діаграму сутностей, матрицю зв'язків і список атрибутів.

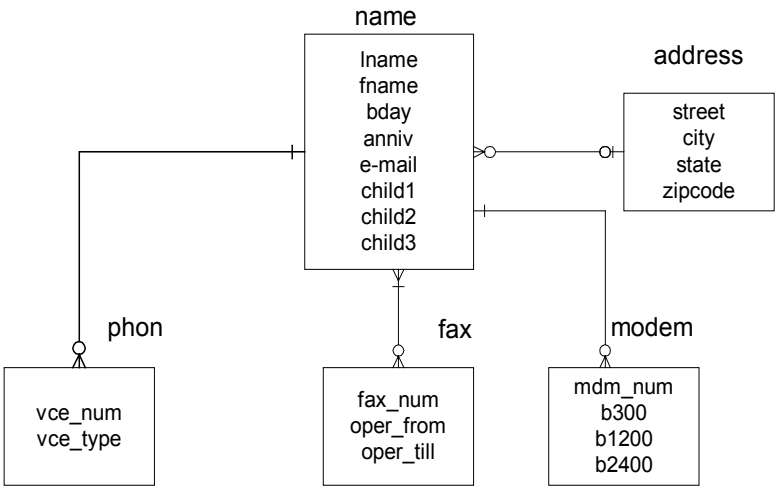


Рис. 2. E-R діаграма телефонного довідника

1.3. Перетворення концептуальної моделі даних у реляційну модель

Основним елементом реляційної моделі є відношення, яке зображується двовимірною таблицею. Імена стовпчиків таблиці є атрибутами відношення. Відношення з множиною його атрибутів утворюють схему відношення, яка зображується іменем відношення та взятим в круглі дужки переліком атрибутів. Наприклад, схема відношення address має вигляд:

address (street, city, state, zipcode) .

Множина схем відношень у проекті має назву "схема реляційної БД". Рядки відношень, які складаються зі значень атрибутів, називаються кортежами. Один кортеж не може двічі входити в дане відношення. Атрибути відношення повинні мати ті самі властивості, що й атрибути сутності, визначені в лабораторній роботі 1. Вони мають бути компонентом елементарного типу.

Порівнюючи властивості відношення й сутності, бачимо, що вони ідентичні. Тому сутності E-R діаграми без змін перетворюємо у відношення реляційної схеми.

Обмеження цілісності в середовищі системи управління базами даних

Під час визначення таблиці командою CREATE TABLE накладаються обмеження на значення в кожному стовпці. Домен, котрий описує обмеження цілісності, містить такі характеристики:

- типи даних (INTEGER, CHAR, DATE та ін.);
- формат (наприклад, yy-mm-dd);
- діапазон значень або значення (наприклад, номер телефону);
- можливі значення або унікальність;
- підтримка NULL значень.

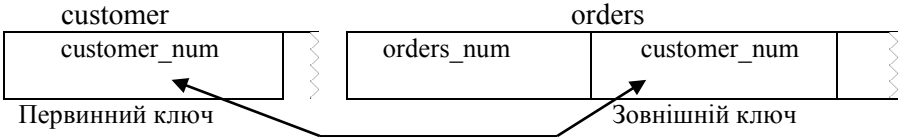
Визначення ключів таблиці

Стовпці таблиці мають ключовий стовпчик або дескриптор. Ключ повинен бути таким, щоб однозначно ідентифікувати рядки. Основними типами ключів є первинний (PRIMARY) і зовнішній (FOREIGN). Первинний ключ таблиці – це стовпець, значення якого різні в кожному рядку. Якщо такого стовпця не існує, то первинний ключ складають зі значень двох або більше стовпців, так, щоб складені значення були різні в різних рядках. Первинний ключ, як правило, належить до цифрового або символьного типу даних (INT, SMALLINT, SERIAL, CHARACTER).

Особливим випадком первинного ключа є ключ, що системно привласнюється (спеціальний номер, що привласнюється кожному рядку сервером БД і має тип SERIAL).

Зовнішнім ключем є стовпець або група стовпців у таблиці, які містять значення первинного ключа іншої таблиці. Зовнішній ключ використовується для створення з'єднання таблиць.

Зразком може служити з'єднання таблиць customer та orders через ключ customer_num :



Зовнішні ключі обмежують можливості видалення рядків з таблиць, що підвищує ймовірність збереження цілісності БД. Після аналізу атрибутів таблиць телефонного довідника і вибору ключів для кожної з них отримаємо таку таблицю:

name	Address	phon	fax	modem
rec_num PK lname fname bday anniv email child1 child2 child3	id_num PK rec_num FK Street State Zipcode	vce_num PK rec_num FK vce_type	fax_num PK rec_num FK oper_from oper_till	mdm_num PK rec_num FK b-300 b-1200 b-2400

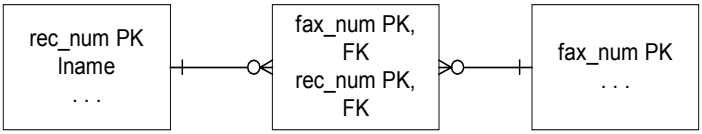
Для таблиці name первинним ключем обраний ключ rec_num, що системно генерується (тип SERIAL), оскільки жоден з атрибутів на цю роль не підходить. Цей ключ можна використати для створення з'єднань з іншими таблицями. Для таблиці address як первинний ключ обраний id_num, який також системно генерується. Для з'єднання таблиці address з name передбачений зовнішній ключ rec_num. У інших таблицях як зовнішній ключ взятий rec_num для забезпечення з'єднання таблиць.

Дослідження і спрощення зв'язків

Зв'язки між двома відношеннями m:n є складними для реалізації в БД, тому їх необхідно спростити. Такий зв'язок можна перетворити на зв'язок 1:n з утворенням нового відношення. Нове відношення звичайно містить атрибути обох пов'язаних відношень. Розглянемо як приклад спрощення зв'язку між name і fax, який є зв'язком m:n. Для розв'язання цього зв'язку створимо нове відношення faxname, яке утворюється при перетинанні початкових відношень. Нове відношення складається з двох атрибутів fax_num і rec_num, які й стають первинним ключем.

Зв'язок між name і faxname 1:n, тому що кожний name може бути пов'язаний з багатьма fax, з іншого боку, кожна faxname комбінація може бути пов'язана з одним rec_num.

Зв'язок між fax і faxname 1:n, тому що кожний fax_num може бути пов'язаний з багатьма faxname комбінаціями. Зв'язки між відношеннями faxname, name і fax мають вигляд:



Після спрощення зв'язків m:n необхідно дослідити й спростити інші особливі зв'язки. До них належать комплексні, рекурсивні й надлишкові.

Комплексний зв'язок - це асоціація між трьома або більше відношеннями. Для розв'язання комплексності треба перевизначити всі комплексні зв'язки, через деяке відношення, пов'язане через бінарні зв'язки з кожним із первинних відношень.

Рекурсивний зв'язок - це асоціація між примірниками одного й того самого відношення. Прикладом такого зв'язку є організаційна структура, коли одні підприємства керуються іншими. Надлишковий зв'язок виникає, коли два або більше зв'язків використовуються для зображення однієї й тієї самої концепції.

Нормалізація моделі даних

Побудовану модель телефонного довідника можна реалізувати в БД. Однак під час її експлуатації можуть виникнути проблеми. Для поліпшення моделі необхідно провести її нормалізацію, яка має на меті:

- зробити модель більш гнучкою;
- усунути надмірність даних;
- підвищити програмну ефективність;
- максимізувати стійкість структури даних.

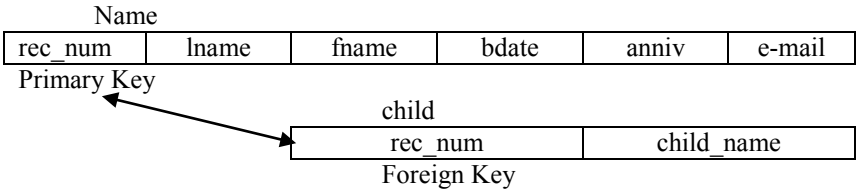
Нормалізація складається з декількох етапів приведення відношень до більш прийнятної фізичної форми.

Перша нормальна форма (1НФ)

Відношення знаходиться у першій нормальній формі, якщо воно не містить груп, що повторюються. В реляційній термінології це означає, що таблиця не повинна містити стовпців, що

повторюються. Стовпці, що повторюються, роблять таблицю менш гнучкою, бо при цьому виникають труднощі під час пошуку даних.

Прикладом стовпців, що повторюються, є стовпці child1, child2, child3 в таблиці name. Для усунення цього розділимо таблицю name на дві, як показано нижче.



Оскільки child не може існувати без зв'язку з таблицею name, то посилаємося до name таблиці за допомогою зовнішнього ключа rec_num. У таблиці modem є стовпці, що повторюються: 300, 1200, 2400. Для нормалізації цієї таблиці формуємо додатковий атрибут b_type. Цей атрибут може містити примірники 300, 1200, 2400 тощо. Модель телефонного довідника, яка приведена до першої нормальної форми, зображена на рис. 3.

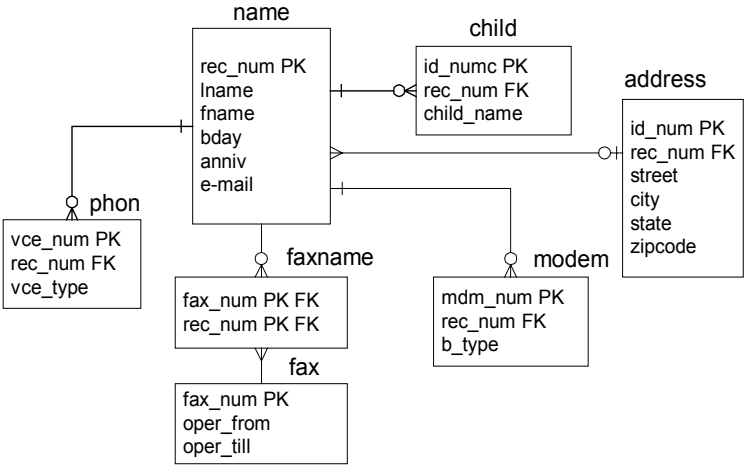


Рис. 3. Модель даних телефонного довідника в 1НФ

Друга нормальна форма (2НФ)

Деяке відношення знаходиться у другій нормальній формі, якщо воно знаходиться в першій і всі його атрибути залежать від первинного ключа. У реляційних термінах це означає, що кожний стовпець у таблиці повинен функціонально повно залежати від первинного ключа таблиці. Те що атрибут залежить від стовпця означає, що якщо значення в стовпці зміниться, то і значення атрибута повинно змінитися. Якщо таблиця має одноколонний первинний ключ, то атрибут повинен залежати від цього ключа. А якщо ключ складений, то неключовий атрибут повинен функціонально залежати від усього ключа, але не знаходитися в функціональній залежності від якоїсь частини ключа.

Для приведення таблиці до 2НФ необхідно:

1) побудувати її проекцію, виключивши атрибути, які не знаходяться в повній функціональній залежності від ключа;

2) побудувати додатково одну або декілька проекцій на частину складеного ключа та атрибути, котрі функціонально залежать від цієї частини ключа.

Третя нормальна форма (3НФ)

Таблиця знаходиться у третій нормальній формі, якщо вона знаходиться у другій нормальній формі і кожний неключовий атрибут нетранзитивно залежить від первинного ключа. Якщо X, Y, Z – атрибути деякої таблиці і існують функціональні залежності $X \Rightarrow Y$, $Y \Rightarrow Z$ і відсутні зворотні залежності $Z \not\Rightarrow Y$, $Y \not\Rightarrow X$, то кажуть, що Z транзитивно залежить від X. Для перетворення таблиці до 3НФ необхідно також побудувати декілька її проекцій.

Отже, в процесі приведення відношень до 2НФ і 3НФ, кількість відношень у моделі БД збільшується. Поява нових відношень зумовлює виникнення проблем підтримки семантичної цілісності моделі.

2. Реалізація моделі даних в середовищі *системи управління базами даних.*

Першим кроком реалізації реляційної моделі даних є визначення доменів обмежень на значення атрибутів у стовпцях таблиць. На другому кроці модель даних реалізується з використанням засобів мови SQL.

Визначення доменів обмеження цілісності даних і встановлення значень атрибутів

Для реалізації реляційної моделі даних необхідно визначити домен кожного стовпця таблиці. Домен стовпця описує обмеження цілісності і встановлює дійсні значення атрибутів. Це необхідно для гарантії семантичної цілісності даних моделі й забезпечення правильного відображення предметної області.

Стовпець доменів специфікує такі обмеження:

1) тип даних;
2) значення за умовчанням;
3) контрольні обмеження.

Крім цього в доменах можуть бути розміщені обмеження для визначення первинного й зовнішнього ключів кожної таблиці.

Типи даних

Першим специфікатором у домені є тип даних, які містить даний стовпець. Тип даних повинен вибиратися зі списку типів, що підтримуються сервером БД.

Обмеження

Після визначення типу даних задаються обмеження на значення частини домену.

Обмеження на дані записуються звичайною мовою. Наприклад, «повинне бути більше ніж нуль», «містить тільки друкарські символи». Для більшої точності синтаксис завдання обмежень може бути взятий такий, як у виразі WHERE. Наприклад, обмеження для домену цілого типу може бути задане так: customer_num >= 5000 AND customer_num <= 99999.

У виразі для завдання обмеження на символьний домен можна використати предикат LIKE. Наприклад, обмеження на домен phone_number можна записати у вигляді:

phone_number LIKE ‘[2-9][2-9][0-9]-[0-9][0-9][0-9]’

Специфіковані домени

Кінцевим кроком у створенні моделі даних є створення специфікованого домену. Домен складається з таких стовпців:

1) ім'я, посилання;
2) тип даних;

3) додаткові правила обмеження даних, записані у вигляді виразу WHERE або на англійській мові.

Один домен може бути використаний до декількох стовпців.

Наприклад, у моделі телефонного довідника телефонний номер входить у декілька стовпців:

Domain	Data Type	Other constraints
address code	SERIAL	1000<code AND code<10000
birthday	DATE	birthday <= TODAY
city	VARCHAR(40,10)	
postal code	CHAR(10)	Only valid postal code

3.Реалізація моделі даних засобами мови SQL

Створення бази даних

Якщо модель даних повністю визначена, можна розпочати створення БД, заснованої на побудованій моделі. База даних містить всі елементи моделі даних, такі як таблиці, представлення, індекси, синоніми та інші об'єкти.

Для створення БД з ім'ям TELS необхідно ввести команду: CREATE DATABASE TELS;

За цією командою у поточній директорії буде створена піддиректорія database_TELS.dbs, у якій буде розміщуватися БД. Для того, щоб розмістити БД в іншій директорії, наприклад, у STUDENT необхідно ввести команду:

CREATE DATABASE TELS IN STUDENT;

Під час створенні БД необхідно визначити тип журналізації, що забезпечує відновлення БД після збоїв. Типи журналізації, що підтримуються, і способи їх завдання різні для різних СУБД. Тут розглядаються варіанти розв’язання цієї задачі в середовищі Informix OnLine Dynamic Server.

Без журналізації

CREATE DATABASE TELS WITH NO LOG;

Цей варіант не рекомендується використовувати, оскільки внаслідок збою, всі зміни, починаючи з моменту отримання останньої архівної копії, будуть втрачені. При виборі такого

варіанта журналізації використовувати команди управління транзакціями, такі як begin work, rollback work тощо не можна.

Регулярна (небуферизована) журналізація

CREATE DATABASE TELS WITH LOG;

Цей варіант рекомендується для більшості БД. У разі збою системи втрачаються тільки результати незавершених транзакцій.

Буферизована журналізація

CREATE DATABASE TELS WITH BUFFERED LOG;

У випадку "краху" системи можлива відсутність яких-небудь втрат або втрачаються самі останні зміни. Ефективність операцій зміни при буферизації цього типу дещо підвищується. Тому такий тип журналізації бажано використовувати в БД, які часто оновлюють.

ANSI-сумісна журналізація

CREATE DATABASE TELS WITH LOG MODE ANSI;

Це те саме, що і регулярна журналізація, але із примусовим виконанням правил ANSI під час обробки транзакцій.

Створення таблиць

Для створення таблиць у спроектованій моделі даних використовується оператор CREATE TABLE. Цей оператор має досить складну форму, але складається переважно з переліку імен стовпців таблиці із наданням такої інформації для кожного стовпця:

- ім'я стовпця;
- тип даних;
- обмеження PRIMARY KEY, якщо стовпець (або стовпці) є первинним ключем;
- обмеження FOREIGN KEY, якщо стовпець (або стовпці) є зовнішнім ключем;
- обмеження NOT NULL, якщо стовпець не повинен містити NULL-значення;
- обмеження UNIQUE, якщо в стовпці не дозволена наявність дублікатів;

- обмеження DEFAULT, якщо стовпець має значення за умовчанням;

- обмеження CHECK, якщо для стовпця існують значення для перевірки.

Отже, команда CREATE TABLE є “словесним” описанням таблиці, побудованої в моделі даних.

Наведемо команди для створення таблиць, що входять до складу моделі даних телефонного довідника, що побудована в попередній роботі.

```
CREATE TABLE name
(   rec_num         SERIAL PRIMARY KEY,
    lname           CHAR(20),
    fname           CHAR(20),
    bdate           DATE,
    anniv           DATE,
    email           VARCHAR(25)
    );
CREATE TABLE child
(   id_numc         SERIAL PRIMARY KEY,
    child           CHAR(20),
    rec_num         INT,
    FOREIGN KEY (rec_num) REFERENCES NAME(rec_num) );
CREATE TABLE address
(   id_num          SERIAL PRIMARY KEY,
    rec_num         INT,
    street          VARCHAR(50,20),
    city            VARCHAR(40,10),
    state           CHAR(5) DEFAULT ‘UA',
    zipcode         CHAR(10),
    FOREIGN KEY (rec_num) REFERENCES name(rec_num)
);
CREATE TABLE phon
(   vce_num         CHAR(13) PRIMARY KEY,
    vce_type        CHAR(10),
    rec_num         INT,
    FOREIGN KEY (rec_num) REFERENCES name (rec_num)
);

CREATE TABLE fax
(   fax_num         CHAR(13),
```

```
    oper_from       DATATIME HOUR TO MINUTE,
    oper_till       DATETIME HOUR TO MINUTE,
    PRIMARY KEY (fax_num)
);
```

CREATE TABLE faxname

```
(   fax_num         CHAR(13),
    rec_num         INT,
    PRIMARY KEY (fax_num, rec_num),
    FOREIGN KEY (fax_num) REFERENCES fax(fax_num),
    FOREIGN KEY (rec_num) REFERENCES name(rec_num)
);
CREATE TABLE modem
(   mdm_num         CHAR(13) PRIMARY KEY,
    rec_num         INT,
    b_type          CHAR(5),
    FOREIGN KEY(rec_num) REFERENCES name (rec_num)
);
```

Зауважимо, що черговість створення таблиць має значення. Так, наприклад, не можна створити таблицю address до створення таблиці name, оскільки в таблиці address є посилання на таблицю name.

4. Використання мови SQL в прикладних програмах

Вставлення операторів SQL в програми

У прикладних програмах, пов'язаних з операціями маніпулювання даними, можна використовувати всі оператори мови SQL. Для цього оператори SQL вбудовуються в початкову програму, написану на одній з мов програмування (C, C++, Pascal, Ada, COBOL та ін.). Існує два способи вбудовування (embedded) операторів SQL. Найбільш простим є вставлення. При цьому оператори SQL є складовою частиною тексту початкової програми. Цей спосіб називається статичним вкладенням.

У деяких програмах виникає необхідність складання операторів SQL по ходу виконання програми, залежно від вхідних даних або умов, що реалізувалися. В цьому випадку використовується динамічне вкладення операторів SQL.

Оператори, що динамічно вбудовуються, не є частиною початкової програми, а формуються в процесі її виконання.

Програмні змінні, що використовуються у вбудованих операторах SQL, називаються головними або host-змінними. Головні змінні використовуються як літерні значення, та повинні бути визначені в програмі. Тип host-змінної не обов'язково повинен збігатися з типом значення в операторі SQL. Ядро БД здійснює необхідне перетворення типів даних.

Для позначення головних змінних у програмі на ESQL в розділі опису може використовуватися знак долара або організовуватися секція оголошення змінних, яка починається рядком BEGIN DECLARE SECTION і закінчується рядком END DECLARE SECTION. Приклад оголошення змінних в ESQL/C:

```
EXEC SQL BEGIN DECLARE SECTION;
 int i, desc_count;
 char demoquery [80],queryvalue [2], char result [NAME_LEN+1];
EXEC SQL END DECLARE SECTION;
```

Для позначення головної змінної в операторі SQL використовується знак долара або знак двокрапка. Оператор SQL в програмі також повинен бути виділений знаком долара, або ключовими словами EXEC SQL.

Вибірка одного рядка даних

Вибірка одного рядка з таблиці виконується вбудованим оператором SELECT.

Головна змінна, в яку вміщується вибраний рядок, вказується за допомогою специфікатора INTO :

```
$ SELECT    avg(total_price)
    INTO  $avg_price  FROM item   WHERE order_num  IN
   ( SELECT order_num FROM orders WHERE order_date <
      DATE(“06/01/96”) );
```

Тут специфікатор INTO вказує головну змінну avg_price, в яку повинні вміщуватися отримані дані. Внаслідок запиту буде виданий рядок даних, що містить один стовпець, і його значення буде вміщене в головну змінну avg_price.

Головні змінні можуть використовуватися не тільки як одержувачі вироблених даних, але і в специфікаторі WHERE.

```
EXEC SQL  SELECT  paid_date
```

INTO :op_date FROM orders
WHERE order_num = :the_order;

Тут використовуються дві головні змінні: op_date для розміщення результату запиту і the_order в специфікаторі WHERE.

У специфікаторі INTO необхідно вказувати головні змінні таким чином, щоб у них розмістився весь створений рядок даних.

Якщо внаслідок запиту буде виготовлено декілька рядків, то ядро БД поверне код помилки.

У БД можуть зберігатися невизначені, тобто так звані NULL-значення. Але, оскільки в мовах програмування вони не обробляються, то при їх передачі з БД у програму може виникнути помилка. Тому в ESQL для розпізнавання NULL-значень використовуються індикаторні змінні, які теж є host-змінними. Ядро бази даних перевіряє чи не є воно NULL-значенням. Якщо це так, то індикаторній змінній буде привласнено значення -1.

Вибірка декількох рядків

Якщо внаслідок запиту до БД може бути повернено більше за один рядок даних, то він повинен оброблятися в два етапи:

- на першому етапі дається старт запиту;
- на другому етапі здійснюється вибірка рядків даних по одній за запит.

Ці операції здійснюються з використанням спеціального об'єкта, що зветься курсором. Розглянемо послідовність операцій, які необхідно реалізувати в програмі для вибірки декількох рядків даних. Всі приклади розглянуті для мови ESQL/C.

Оголошення курсору і пов'язаного з ним оператора SELECT здійснюється таким чином:

DECLARE the_item CURSOR FOR SELECT fname, lname
INTO :f_name, :l_name FROM customer WHERE lname > “K”;

Наступним кроком є відкриття курсору, що фактично приводить до початку виконання оператора SELECT.

OPEN the_item;

Для вибірки рядка результату використовується оператор FETCH. Він іменує курсор, а також задає імена головних змінних для прийому даних. Після прочитання останнього рядка даних

курсор необхідно закрити. Нижче наведений приклад на ESQL/C, що демонструє подальші кроки цього процесу:

```
FETCH the_item;
IF ( strncmp (SQLSTATE, "00", 2) != 0 )
  break; /* error */
THEN  display ($f_name, $l_name);
```

Тут SQLSTATE - це масив області зв'язку ядра бази даних і ESQL/C, значення елементів якого відображають успішність виконання операторів SQL. Зазначимо, що специфікатор INTO може розміщуватися в операторі SELECT, або в FETCH.

Курсор може працювати в одному з двох режимів: послідовному або такий, що прокручує (scroll). При виборі рядків через послідовний курсор для кожного запиту необхідно відкрити свій курсор, і внаслідок одного звертання вибирається один рядок. У розглянутому прикладі був використаний послідовний курсор.

При вибірці через scroll-курсор, у разі одного відкриття курсору можна читати багато рядків. Такий курсор оголошується за допомогою ключового слова SCROLL, як показано в прикладі:

```
EXEC SQL DECLARE  the_item  SCROLL CURSOR;
```

Для scroll-курсору можна використати різноманітні додаткові специфікації курсорної вибірки. Специфікація ABSOLUTE, наприклад, визначає порядковий номер для курсорної вибірки:

```
EXEC  SQL  FETCH  ABSOLUTE  numrow  the_item  INTO
:f_name, :l_name;
```

Цей оператор прочитує рядок, позиція якої задана у головній змінній numrow. Таким чином, у разі використання послідовного курсору при кожному виконанні оператора FETCH, ядро БД повертає вміст поточного рядка і знаходить наступний рядок. А у разі використання scroll-курсору ядро БД створює активну множину рядків у тимчасовій таблиці, що прискорює доступ до даних.

Модифікація даних у ESQL-програмах

Видалення рядків

Рядок у таблиці можна видалити за допомогою оператора DELETE. При цьому рядок, що видаляється, може бути вибраний

за допомогою специфікатора WHERE або через курсор. У першому випадку реалізовується пряме видалення:

EXEC SQL DELETE FROM items WHERE order_num = :o_num;

Тут оператор DELETE працює безпосередньо з базою даних. Після завершення оператора інформація про його виконання вміщується в область зв'язку SQLCA. Навіть при появі помилки третій елемент масиву SQLERRD містить кількість видалених рядків. Значення змінної SQLCODE вказує характер завершення операції, і негативне значення відповідає відсутності помилки.

Якщо рядки, що видаляються оператором DELETE, не можуть бути задані за допомогою специфікатора WHERE, то для вибірки рядків, що видаляються, можна використати курсор:

```
EXEC SQL BEGIN DECLARE SECTION;
int  ord_num, ord_date;
int  dup_cnt, ret_code;
EXEC SQL END DECLARE SECTION;
$ DECLARE scan_ord CURSOR FOR SELECT order_num, order_date
INTO :ord_num, :ord_date FROM orders for UPDATE; /* курсор може
використовуватись для модифікації даних */
$ OPEN scan_ord;
if (sqlca.sqlcode != 0)          /*неуспішне завершення open */
  return (sqlca.sqlcode);
$ begin work;                    /* початок транзакції */
  dup_cnt = 0;                   /* значення за умовчанням */
for ( ; ; )
{ $ FETCH  scan_ord;             /* наступний рядок */
 if (sqlca.sqlcode != 0) break;          /*вихід, якщо неуспішна вибірка
рядка*/
    $ SELECT count(*) INTO :dup_cnt FROM orders WHERE
    order_num = :ord_num;
 if (dup_cnt > 1)
  { $ DELETE WHERE current of scan_ord;
   /*видалити рядок, на який вказує курсор */
   if( sqlca.sqlcode != 0) { ret_code = sqlca.sqlcode; break; }
  }  }  /* end for ( ; ; )*/
if (ret_code = = 100)/* кінець даних*/
  $ commit work;                 /* реалізувати зміни в БД */
 else
  $ rollback work;               /* відкат транзакції */
return (ret_code); }
```

Ця програма видаляє рядки, що містять дублікати номерів замовлень. Якщо курсор відкритий правильно, програма починає транзакцію і проходить у циклі по рядках таблиці. Для кожного рядка виконується вбудований SELECT, щоб визначити скільки рядків містить номер замовлення такий самий, як і поточний.

Вставка рядків

Вставку рядків у таблицю можна виконувати з використанням курсора. Курсор для вставки оголошується також, як і для вибірки: EXEC SQL DECLARE new_custs CURSOR FOR INSERT INTO customer (company, fname, lname) VALUES (the_company, the_fname, the_lname);

Під час відкриття курсора new_custs в пам'яті створюється буфер для зберігання блоку рядків. Буфер приймає рядки даних по мірі видачі їх програмою за командою PUT new_custs; і, після заповнення буфера, блок передається ядру БД. По закінченні використовується оператор FLUSH для дозапису рядків, що залишилися в буфері, після чого можна завершувати дану транзакцію.

Зауважимо, що всю конструкцію з курсором можна викинути, а замість PUT поставити INSERT. Результат операції буде таким самим. Однак використання курсора для вставки дозволяє виконувати цю операцію значно швидше

Додаток 1.
ОПИСИ ПРЕДМЕТНИХ ОБЛАСТЕЙ ІНФОРМАЦІЙНИХ СИСТЕМ

Варіант №1

База даних клієнтів - позичальників банку.

БД повинна забезпечити:

- ведення кредитних історій позичальників;
- дані про всі транзакції по погашенню кредиту;
- дані по нарахуванню відсотків на невиплачений залишок кредиту;
- дані про фінансову спроможність клієнта та нерухоме майно.

На кожного позичальника заводиться кредитна історія,в якій вказується номер, сума позики , дата видачі,дати та суми щомісячних погашень боргу,залишок боргу. На залишок боргу щомісяця нараховуються відсотки. Ведеться баланс рахунку клієнта, де вказується сплачена на

поточний момент сума за кредит і несплачений залишок. Окремо зберігаються дані про нерухоме та рухоме майно, це назва ,оціночна вартість,чи не знаходиться під заставою.

Варіант №2

База даних бібліотеки.

БД повинна забезпечити :

- ведення автоматизованого обліку(видачі\повернення) книг;
- ведення черг на літературу (за замовленнями);
- врахування рейтингу видань (кількість читачів, дата останньої видачі;
- складання списку боржників по рокам.

База даних використовується для ведення обліку книг, та пошуку книг, замовлених читачами. Зберігається інформація про всі книги, це назва ,автор, код, об'єм, і т.д.. Ведеться облік виданих екземплярів книг, кому видано, дата видачі і дата повернення. Бібліотека має декілька відділів, до яких належать книги. Зберігається також інформація про читачів, дані паспорта, № - читацького квитка та інше.

Варіант № 3

База даних складу.

БД повинна забезпечити:

- дані про наявний товар на складі;
- дані по отриманню.та відвантаженню товарів по кожному постачальнику, та отримувачу;
- дані по оплаті отриманого товару.

На склад поступають деякі товари, а також вони відвантажуються в магазини. На отриманий ,або відвантажений товар оформляється документ.

Оплата товару здійснюється попередньо по платіжному документу, або на виплату

Варіант № 4

База даних поліклініки.

БД повинна забезпечити:

- ведення медичних карток пацієнтів;
- врахування рецептів,направлень на аналізи, процедур;
- врахування платних послуг з видачею рахунків;

В базі даних зберігаються дані пацієнта, такі як: прізвище, адреса, рік народження, місце і посада на роботі, медичні дані, такі як група крові, перенесені захворювання та інші. Зберігаються дані про всі звернення до лікаря, дата, прізвище та фах лікаря , діагноз та призначення лікування, а

також результати лікування. Окремо зберігаються дані про перебування пацієнта в лікарні, дата, назва лікарні, адреса, прізвище лікаря, діагноз, призначення лікування.

Варіант № 5

База даних охоронного підприємства.

БД повинна забезпечити :

- отримання інформації про спрацювання систем сигналізації;
- дані про виїзди на місце подій чергових нарядів та результати;
- дані по особовому складу;
- дані по наявних транспортних засобах.

В базі даних зберігаються дані про об'єкти, які охороняються, це адреса, назва об'єкта, тип сигналізації, відстань, також зберігаються дані про технічні характеристики систем сигналізації, які використовуються. Зберігаються дані про наявні авто та їх характеристики, такі як марка,швидкість, та інше. В базі даних фіксуються всі спрацювання охоронної системи, та результати виїзду на місце події співробітників,дата і час ,об'єкт,хто виїздив на виклик і результат.

Варіант № 6

База даних компанії по виконанню проектів.

БД повинна забезпечити:

- контроль стану виконання проектів по кожному підрозділу ;
- об'єм виконаної роботи (в годинах) кожним співробітником по кожному проекту;
- нарахування оплати роботи співробітникам за місяць.

В базі даних зберігаються дані про співробітників, такі як прізвище, адреса, анкетні дані – кваліфікація, посада ,підрозділ .та інше. На виконання роботи видається наряд ,в якому вказані дані співробітника ,номер проекту,вид роботи, час виконання, дати початку і закінчення,вартість роботи. Виконання роботи оформляється актом приймання. Співробітник приймає участь у виконанні декількох проектів.

Варіант № 7.

База даних підприємства з виробництва меблів.

БД повинна забезпечити:

- дані по виконаним замовленням;
- дані по замовленням , які виконуються;
- замовлення на комплектуючі ;
- дані про оплату.

Підприємство виробляє меблі по замовленнях. В замовленні вказуються дані замовника,
прізвище, адреса ,вид меблів,вартість,вид оплати. На кожен вид меблів підприємство замовляє комплектуючі на інших підприємствах. Замовлення на комплектуючі формується на основі отриманих замовлень. Таким чином необхідно вести облік отриманих замовлень, номенклатури комплектуючих ,формувати замовлення на комплектуючі.

Варіант№8

База даних транспортного підприємства.

БД повинна забезпечити:

- дані по виконаних замовленнях за певний проміжок часу;
- дані про задіяні ,та незадіяні транспортні засоби в даний час;
- звіт про виконану водіями роботу за певний проміжок часу;

Транспортне підприємство виконує замовлення на перевезення вантажів. В замовленні вказано прізвище замовника,дата, адреса, адреси отримання вантажу і вивантаження, відстань, характер вантажу ,вартість, а також статус «не виконано» ,або «виконано» На основі отриманих замовлень складається план їх виконання на день,з врахуванням наявних транспортних засобів. Для виконання замовлення призначається певний транспортний засіб і водій,на які виписується маршрутний лист ,в якому крім інформації вказаній в замовленні ,є відмітка про виконання

замовлення. В кінці дня підводяться підсумки виконання замовлень. Зберігається також інформація про транспортні засоби,де вказується тип, номер,статус, тобто вільний, чи призначений.

Варіант № 9.

База даних будівельної компанії.

БД повинна забезпечити:

- дані про стан будівництва по кожному об'єкту;
- інформацію про витрачені кошти по кожному об'єкту;
- дані про необхідні матеріали на поточний період (місяць).

Будівельна компанія будує декілька об'єктів. На кожен обєкт є проектна документація, в якій зазначено тип об'єкту, адреса, дати початку і закінчення, вартість, відповідальний виконавець. До проекту додається кошторис, в якому розписані об'єми робіт по кожному виду,вартість матеріалів. Кожен місяць складається акт на виконані роботи по кожному об'єкту,кожному виду робіт ,витрачені матеріали,витрати коштів, дата. Необхідно мати інформацію про стан будівництва по кожному виду робіт (в відсотках),по кожному обєкту.

Варіант № 10.

База даних підтримки клієнтських операцій в банкоматах.

БД повинна забезпечити:

- інформацію про власників карток;
- інформацію про поточний стан рахунків;
- дані про виконані операції з рахунками.

Клієнт може мати декілька рахунків різних видів.

В базі даних повинні зберігатись повна інформація про клієнтів,прізвище, адреса,№-паспорта, РІН-код. Інформація про пластикові картки (номер картки, дата реєстрації, ідентифікатор банку, PIN – код, тип картки) .Дані про рахунок клієнта з вказаними номером, датою, наявні кошти (баланс), тип рахунку. Зберігаються також дані про виконані транзакції (операції) по зняттю коштів ,зарахуванню, та переведенні на інші рахунки.

Варіант №11

БД акціонерного товариства.

БД повинна забезпечити:

- інформацію про власників акцій;
- дані про акції (одною акцією можуть володіти декілька акціонерів);
- інформацію про прибутки підприємств та нараховані дивіденди.

Фізичні особи володіють акціями підприємств. З прибутку підприємств на акції нараховуються дивіденди в кінці року. Необхідно мати інформацію про прибутки підприємств для нарахування дивідендів. Для цього зберігаються дані підприємств,такі як назва ,код, адреса прибуток за рік.,Акція характеризується номером, категорією, номіналом ,акціонерним товариством. Власник акції ідентифікується необхідними даними, прізвище,адреса,№-паспорта і ІК. Необхідно також зберігати інформацію про нараховані дивіденди по кожні акції, і кожному власнику.

Варіант №12

БД магазину.

БД повинна забезпечити:

- облік проданої продукції по видам, кількості, і вартості;
- облік всіх виконаних замовлень по клієнтам;
- облік наявної продукції на складі
- формування замовлення на поставку продукції.

Клієнт замовляє деякий перелік продукції менеджеру з продаж, для чого заповнює замовлення, котре складається з номера, дати, прізвища менеджера та списку товарів, вартість. Клієнт оплачує вартість товару готівкою,або кредиткою,і отримує платіжний чек. В свою чергу менеджер замовляє й отримує продукцію зі складу за накладною. В одній накладній може бути вказано декілька видів продукції.

Кожен вид продукції характеризується назвою, кодом, виробником, вартістю. Магазин замовляє продукцію у поставщиків.

Варіант № 13

БД ВИШУ.

БД повинна забезпечити:

- дані про студентів по кожній спеціальності;
- інформацію про викладачів кафедри;
- дані про результати сесії по кожному студенту;

Вищий навчальний заклад має декілька факультетів , на кожному з яких є кафедри., кожна з яких готує фахівців певної спеціальності .

Студент навчається в певній навчальній групі по певній спеціальності. Викладач кафедри веде заняття по одній, або декільком дисциплінам. В кожній групі є куратор , який являється викладачем випускової кафедри. Викладачі кафедри читають певні дисципліни для груп студентів. Дисципліна характеризується назвою,номером навчального плану, номером семестру, і номером групи. Факультет характеризується назвою, ПІБ декана, телефоном деканату. Спеціальність характеризується номером, назвою, номером напрямку, номером ліцензії, кількістю студентів.

Варіант №14

БД «Кафедра»

Завдання-інформаційна підтримка навчального процесу на кафедрі ВНЗ.

БД повинна містити навчальний план, розклад занять ,списки груп ,які спеціалізуються по даній кафедрі.

Для складання розкладу занять по навчальним групам використовується навчальний план.,який розробляється для кожної спеціальності, тобто для тих груп які навчаються на даній спеціальності на всі роки навчання. Кожна дисципліна навчального плану викладається викладачем кафедри. Викладач може викладати одну, або більше дисциплін, а одну дисципліну можуть читати декілька викладачів

БД повинна забезпечити складання:

-розкладу занять по групам для певної спеціальності;
-розклад занять для викладачів кафедри;
.

Варіант № 15

БД дипломних проектів.
БД повинна забезпечувати:
- дані про дипломні проекти,та студентів,які їх виконують
-дані про стан виконання проектів;
- інформацію про керівників;
- інформацію про результати захисту.

Дипломники виконують дипломні проекти на випускаючій кафедрі. Проект має назву,спеціальність , терміни виконання. Дипломник характеризується ПІБ, № групи, темою ,керівником диплому. Керівник характеризується ПІБ, назвою кафедри, вчена ступінь, посада, Кожний керівник може керувати декількома дипломниками. На виконання проекту складається план , в якому вказані назви розділів і дата виконання планована і фактична. Кожного місяця керівник оцінює стан виконання проекту у відсотках.,який заноситься в план,Проект рецензується рецензентом. У висновку рецензент виставляє оцінку,рекомендацію відносно присудження кваліфікації,або ні, і зауваження. Рецензент є викладачем іншої кафедри і характеризується тими ж атрибутами ,що і керівник.

Результати захисту містять оцінку захисту, оцінку рецензента і рекомендацію.

Варіант №16

БД «Магазин».

Завдання- інформаційна підтримка діяльності магазину вибраного профілю.

БД повинна забезпечити:

- облік постачальиків і поставок товарів;
- врахування продажу по відділах;
- підрахунок залишків товарів (по відділах);
- оформлення замовлень на товари ,запаси яких закінчуються;
- підведення фінансових результатів дня (по відділах і по магазину);
- аналіз результатів роботи продавців .

* В одній поставці може бути декілька товарів, причому може бути однаковий товар різних виробників. Товар має штрих код і певну вартість ,і опис.

Варіант №17

БД по нерухомості.

Завдання- інформаційна підтримка діяльності фірми по продажу і здачі в оренду житлових і нежитлових приміщень.

БД повинна забезпечити:

- ведення списку приміщень ,призначених до продажу , або здачі в оренду з їх характеристиками;
- підтримка архіву проданих і зданих в оренду приміщень;
- проводити пошук варіантів у відповідності з вимогами клієнтів.

Клієнт надає замовлення, в якому вказує тип приміщення, і бажані його характеристики, такі ,як площа,поверх, вартість ,та інше.

Необхідно передбачити отримання статистики:

- наявність приміщень різних типів;
- зміна цін на ринку.

Варіант №18

БД «Готель»

Завдання- інформаційна підтримка діяльності готелю.
Готель має номери різного класу по комфортності , і кількості місць в номері, і відповідна вартість за добу..
Клієнт може дати замовлення он-лайн,тобто забронювати номер ,або безпосередньо. В замовленні вказується тип номера і дати заїзду і звільнення номера.
БД повинна забезпечувати :

- ведення списку гостей;
- облік заброньованих місць;
- ведення архіву гостей за останній рік;

Необхідно передбачити:

-отримання списку вільних номерів (по кількості місць і класу);
- отримання списку номерів з датами їх звільнення;
- видача рахунків на оплату номерів і послуг;

Варіант №19

БД «Продаж квитків»

Завдання – інформаційна підтримка діяльності квиткових транспортних кас.
Продаж квитків виконується на певний вид транспорту,на певну дату,на конкретний рейс.
При купівлі квитка пасажир вказує номер рейсу, дату , та бажане місце.
Після купівлі квитка місце відмічається, як зайняте.
БД повинна забезпечити:

- ведення списку рейсів і квитків на них ;
- врахування заброньованих місць ;
- ведення архіву пасажирів;

Необхідно передбачити:

- пошук місць на рейс у відповідності з замовленням пасажира;
- видачу інформації про наявність місць на конкретний рейс;
- продаж квитків в обидва кінці;

- отримання списку проданих місць.

Варіант №20

БД «Ресторан»

Завдання - інформаційна підтримка діяльності ресторану. Клієнти можуть подавати замовлення через Інтернет,або безпосередньо.
БД повинна забезпечити :

- ведення замовлень на поточну дату,а також і архіву замовлень,
- ведення каталогу меню по видам блюд,блюдам,цінам,та розкладкою продуктів ,необхідних для приготування цих блюд;
- каталог оплачених рахунків по виконаним замовленням з даними офіціантів та кухарів;
- обчислення кількості витрачених продуктів ,для формування замовлення на їх поставку;
- формування фінансової звітності по видаткам ,та прибуткам.

Варіант №21

БД «Туристична агенція»

Завдання – інформаційна підтримка діяльності туристичної агенції.
БД повинна забезпечити :

- ведення каталогу наявних турів;
- ведення списку груп, які формуються на відповідні тури;
- дані про оплату, форма оплати,сума , дата;
- ведення архіву виконаних турів.

Потрібно передбачити:

- можливість пошуку вільних місць на певні тури;
- видачу інформації на сайт про тури на які формуються групи в якій міститься маршрут, дати виконання туру, вид транспорту,класність готелю розміщення,харчування,заплановані екскурсії,вартість туру;
- фінансовий звіт про витрати по певними видам;
- дані про екскурсоводів ,виконані ними тури,та відгуки клієнтів.

Варіант №22

\

БД « Білінгова система»

Завдання – інформаційна підтримка фірми по веденню розрахунків за використання комунальних послуг фізичними та юридичними особами.
БД повинна забезпечити:

- ведення переліку тарифів на кожний вид споживаного ресурсу (газ.елекрика,вода,опалення). Тарифи залежать від категорії споживача, та об'єму спожитого ресурсу;
- збір інформації від споживачів по об'єму спожитого ресурсу за попередній місяць ;
- формування рахунків на оплату послуг;
- ведення даних по приладах обліку спожитого ресурсу у споживачів;
- контроль оплати рахунків . Оплата проводиться через банківські установи.

Потрібно передбачити:

- ведення архіву проплат , та використаного ресурсу за рік; формування повідомлень про заборгованість абонентів.

Варіант №23

БД «Склад товарів»

Завдання – інформаційна підтримка діяльності складу товарів (одяг,взуття та інше).
БД повинна забезпечити :

- ведення каталогу наявних товарів на складі. Дані про товар включають назву групу
  характеристики, ціну, поставщик ,№ поставки,дату та інше;
- каталог поставок на склад,та поставки зі складу;
- дані про залишки товарів на складі на поточну дату;
- дані про поставщиків.

Варіант №24

БД клієнтів банку.

БД повинна забезпечити:

- дані про клієнтів банку і про стан їхніх рахунків;
- дані про проведені операції з рахунками;
- інформацію про нараховані відсотки за певний період.

Клієнт має банківські рахунки в різних банках. Рахунок має номер,код банку, дату відкриття,тип рахунку,сума коштів. Клієнт характеризується ПІБ, адресою, № телефону, ідентифікаційним кодом. На рахунок можуть переводитись кошти , або зніматись.

Операції з рахунком визначаються датою, видом операції, розмір коштів ,які знімаються , або нараховуються. На кошти , які знаходяться на рахунку можуть нараховуватись відсотки.Банк характеризується, назвою, адресою, № телефону, № ліцензії.

Завдання №25

БД рекрутингового агенства.

Рекрутингове агентство отримує заявки від організацій про потребу в фахівцях певних професій і кваліфікацій. Агентство підшукує потрібних фахівців за розміщеними в Інтернеті резюме, а також за отриманими заявками від осіб які шукають роботу. Після співбесіди кандидатом агентство видає направлення для працевлаштування.

БД агентства повинна містити:

- дані про вакантні посади;
- дані про пошукачів роботи;
- дані про працевлаштованих;
- необхідна інформація про організації та підприємства.

Варіант № 26

БД обслуговуючої компанії (ЖЕК)

Обслуговуюча компанія надає послуги мешканцям багатоквартирних будинків по ремонту сантехнічного обладнання , водо,тепло постачання та інше. Мешканець звертається з заявкою про надання послуги. Менеджер визначає виконавця робіт і видає наряд на виконання. Виконавець по закінченню робіт підприсує наряд у мешканця. За виконану роботу йому нараховується оплата. Мешканець оплачує послугу.

БД повинна містити :

-заявки на виконання робіт;

- наряди на роботи з відміткою про виконання;

- дані мешканців;

БД повинна забезпечити:

-контроль виконання робіт;

-нарахування оплати працівникам;

- контроль оплати послуги мешканцями.

Варіант № 27

Кожне акціонерне товариство повинно всю документацію про свою діяльність надавати організації – зберігачеві.

БД зберігача АТ.

БД зберігача повинна забезпечувати:

- дані про акціонерні товариства;
- дані про випущені акції ;
- інформація про придбані акції;
- дані про поточні котування акцій на біржі;

Вся документація акціонерного товариства зберігаються у зберігача (організації). В цей перелік входить :

-угоди з АТ про ведення реєстру;

-інформація про акціонерне товариство,назва,код ,адреса, телефон,голова, дата створення, уставний фонд,кількість;

-інформація про акціонерів ,прізвища,код, адреси,номери і вартість акцій, якими вони володіють;

- інформація про емісію (випуски) акцій : номери акцій ,вартість , код власника (якщо вона придбана) ,дата випуску,№-рішення зборів;

- інформація про збори акціонерів : номер протоколу зборів,дата № - рішення.

Варіант №28

БД пункту прокату автомобілів, який має у власності автомобілі певних марок

Клієнт може отримати автомобіль на прокат представивши необхідні документи (паспорт, водійське посвідчення), та оплативши вартість послуги на певний проміжок часу.. Оформляється угода і клієнт отримує автомобіль в користування.

БД повинна містити :

-дані про наявні автомобілі;

- дані про клієнтів;
- угоди про оренду;
- дані про оплату послуги
- інформацію про автомобілі, які знаходяться в оренді (дані автомобіля, дата видачі, дата повернення.

Варіант № 29

База даних біржі цінних паперів.

Акціонерні товариства надають біржі свої акції для продажу. Акції мають певну вартість. На момент купівлі-продажу акції мають іншу вартість (котування). Акції купують і продають від імені акціонерних товариств брокери. Брокер має певну суму коштів на рахунку, на які він може купувати акції.

БД повинна містити:

- дані про наявні акції (кількість, вартість, власник);
- дані про здійснені продажі (покупки) , номери акцій, власник, кількість, дата ,котування на дату покупки ,покупець, брокер;
- дані про брокерів (їх,депозит,рейтинг, особисті дані);
- дані про власників акцій.

Варіант № 30

База даних системи бронювання та продажу квитків на залізничні потяги. Потяги виконують пасажирські перевезення за певними маршрутами.

Потяг, який виконує певний рейс має вагони різних класів ,в яких є певна кількість місць кожне з них має певний номер Після вибору номера рейса, номера вагону , дати, та номера місця пасажир оплачує проїзд . Після цього статус місця змінюється з вільного на зайняте.

В базі даних повинна бути інформація про місця в кожному вагоні рейсового потягу на певні дати , та їх статус. При покупці квитка в базу повинна заноситись інформація про пасажира, та оплату квитка . Повинна також зберігатись інформація про кожний рейс, а саме початкова, кінцева, та проміжні залізничні станції, з відміткою тих , де він здійснює зупинку.