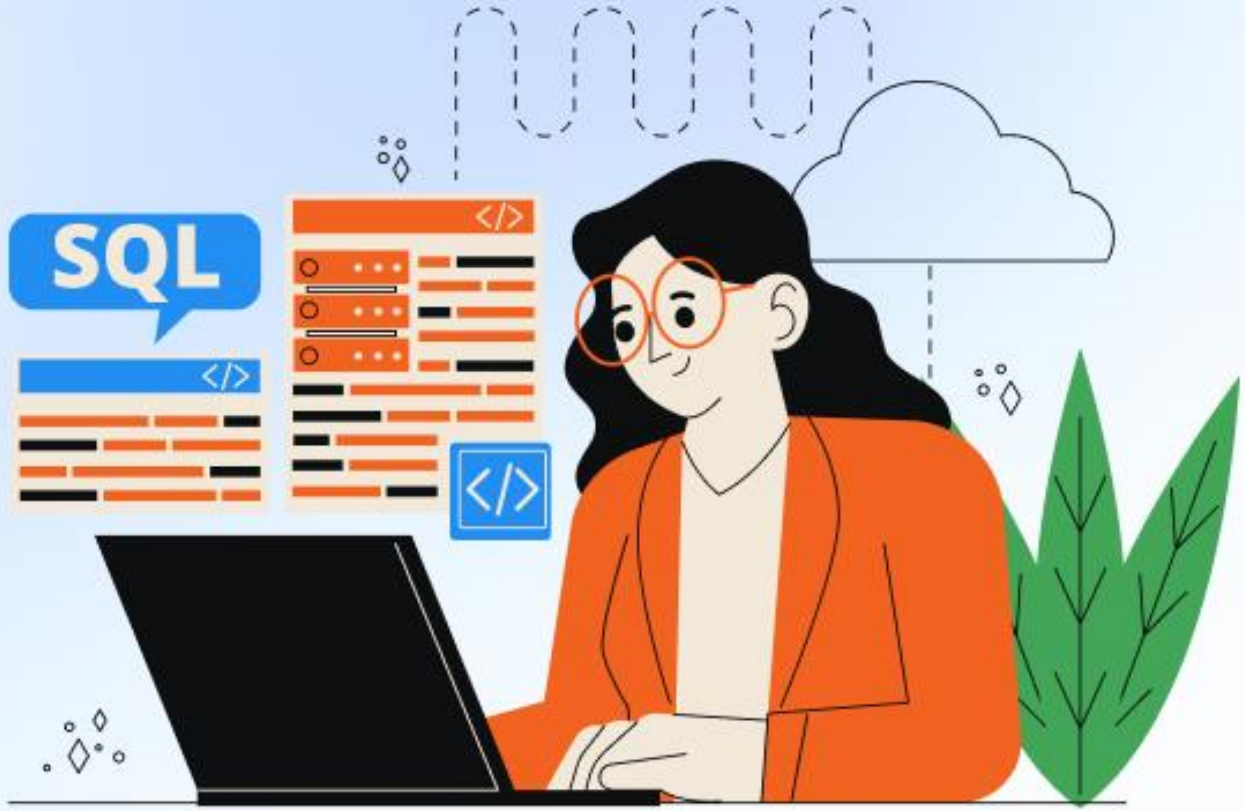
Мова структурованих запитів SQL



Історія та розвиток SQL

SQL (Structured Query Language – Структурована Мова Запитів) є однією з найважливіших технологій у сфері управління базами даних.

Історія SQL починається у 1970 році, коли Едгар Кодд, науковець IBM, опублікував свою революційну роботу "A Relational Model of Data for Large Shared Data Banks".

Історія та розвиток SQL

У 1979 році компанія Relational Software Inc. (пізніше перейменована на Oracle Corporation) випустила свою комерційну реалізацію SQL, яка працювала на основі реляційної моделі даних.

У 1980-х роках IBM продовжила роботу над SQL, впровадивши її у своїх продуктах DB2 та SQL/DS.

Історія та розвиток SQL

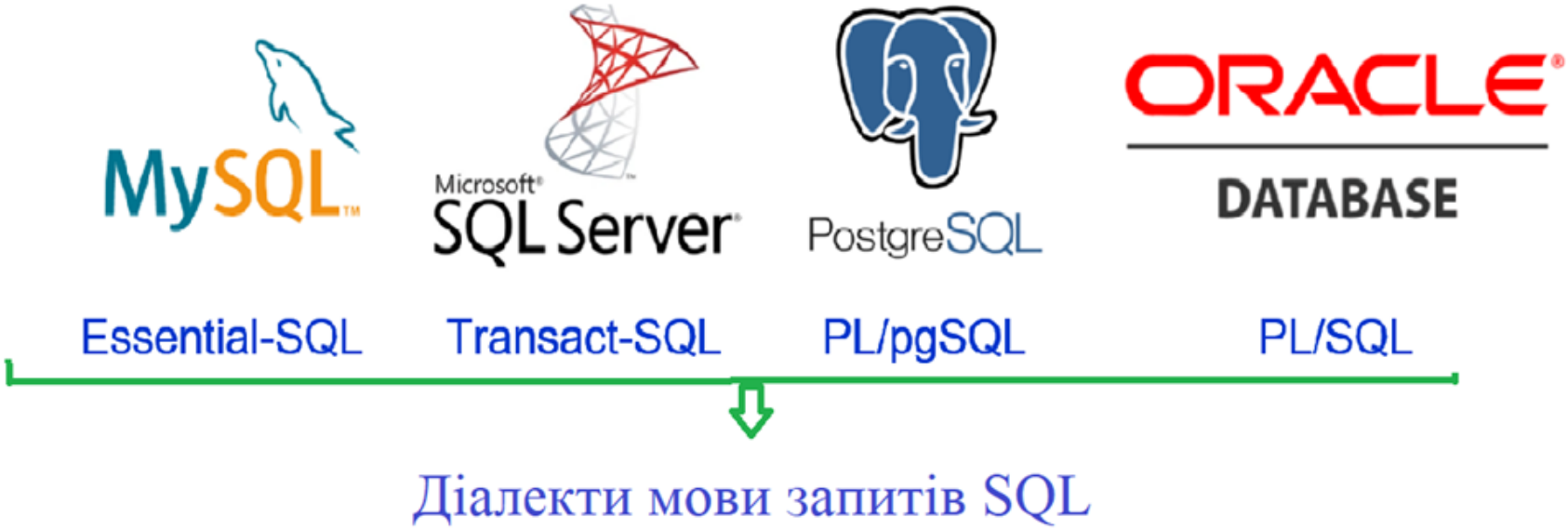
У 1986 році був прийнятий перший стандарт SQL — SQL-86 (також відомий як SQL1).

Наступний великий крок стався у 1992 році, коли був прийнятий стандарт SQL-92. SQL-92 став основою для багатьох сучасних реляційних СУБД.

SQL:2003 додав можливості роботи з XML-даними, що стало важливим кроком у зв'язку зі зростанням популярності веб-технологій.

Останні версії стандарту, такі як SQL:2008, SQL:2011 та SQL:2016, продовжують розширювати функціональність мови, додаючи нові можливості для аналітики, роботи з JSON-даними, а також поліпшення продуктивності.

Історія та розвиток SQL



Ми вивчатимемо мову структурованих запитів SQL на прикладі СУБД Microsoft SQL Server

Теоретичні мови запитів

Мови запитів – це спеціалізовані мови програмування, призначені для управління та маніпуляції даними у базах даних.

Теоретичні мови запитів виникли з необхідності формалізації роботи з даними в реляційних базах. Вони базуються на математичних концепціях, таких як реляційна алгебра та реляційне числення, які були вперше запропоновані Едгаром Коддом у 1970 році

Теоретичні мови запитів

Правила Кодда.

Правило 0: Основне правило (Foundation Rule). Система, яка рекламується або позиціонується як реляційна система управління базами даних, повинна бути здатна управляти базами даних, використовуючи виключно свої реляційні можливості.

Правило 1: Інформаційне правило (The Information Rule). Вся інформація в реляційній базі даних на логічному рівні повинна бути явно представлена єдиним способом: значеннями в таблицях.

Правило 2: Гарантований доступ до даних (Guaranteed Access Rule). У реляційній базі даних кожне окреме (атомарне) значення даних має бути логічно доступно за допомогою комбінації імені таблиці, значення первинного ключа і імені стовпця.

Правило 3: Систематична підтримка відсутніх значень (Systematic T reatment of Null Values). Невідомі, або відсутні значення NULL, відмінні від будь-якого відомого значення, повинні підтримуватися для всіх типів даних при виконанні будь-яких операцій. Наприклад, для числових даних невідомі значення не повинні розглядатися як нулі, а для символьних даних - як порожні рядки.

Теоретичні мови запитів

Правило 4: Доступ до словника даних в термінах реляційної моделі (Active On-Line Catalog Based on the Relational Model). Словник даних повинен зберігатися в формі реляційних таблиць, і СУБД повинна підтримувати доступ до нього за допомогою стандартних мовних засобів, тих же самих, які використовуються для роботи з реляційними таблицями, що містять призначені для користувача дані.

Правило 5: Повнота підмножини мови (Comprehensive Data Sublanguage Rule). Система керування базами даних повинна підтримувати хоча б один реляційний мову, який:

- має лінійний синтаксис,
- може використовуватися як інтерактивно, так і в прикладних програмах,
- підтримує операції визначення даних, визначення уявлень, маніпулювання даними (інтерактивні та програмні), обмежувачі цілісності, управління доступом і операції управління транзакціями (begin, commit і rollback).

Правило 6: Можливість зміни уявлень (View Updati ng Rule). Кожна вистава має підтримувати всі операції маніпулювання даними, які підтримують реляційні таблиці: операції вибірки, вставки, зміни та видалення даних.

Теоретичні мови запитів

Правило 7: Наявність високорівневих операцій управління даними (High - Level Insert, Update, and Delete). Операції вставки, зміни та видалення даних повинні підтримуватися не тільки по відношенню до одного рядка реляційної таблиці, але і по відношенню до будь-якого безлічі рядків.

Правило 8: Фізична незалежність даних (Phys ical Data Independence). Додатки не повинні залежати від використовуваних способів зберігання даних на носіях, від апаратного забезпечення комп'ютерів, на яких знаходиться реляційна база даних.

Правило 9: Логічна незалежність даних (Logical Data In dependence). Представлення даних в додатку не повинно залежати від структури реляційних таблиць. Якщо в процесі нормалізації одна реляційна таблиця розділяється на дві, подання має забезпечити об'єднання цих даних, щоб зміна структури реляційних таблиць не позначалося на роботі додатків.

Теоретичні мови запитів

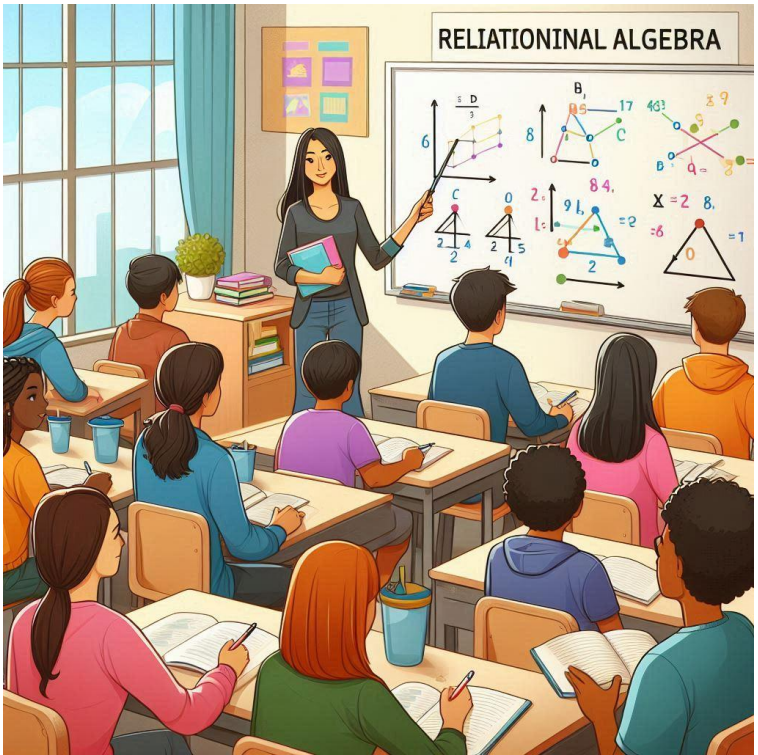
Правило 10: Незалежність контролю цілісності (Integrity Independence). Вся інформація, необхідна для підтримки цілісності, повинна знаходитися в словнику даних. Мова для роботи з даними повинен виконувати перевірку вхідних даних і автоматично підтримувати цілісність даних.

Правило 11: Незалежність від розташування (Distribution Independence). База даних може бути розподіленою, може перебувати на декількох комп'ютерах, і це не повинно впливати на додатки. Перенесення бази даних на інший комп'ютер не повинен впливати на додатки.

Правило 12: Узгодження мовних рівнів (The Nonsubversion Rule). Якщо використовується низькорівнева мова доступу до даних, віна не повинна ігнорувати правила безпеки і правила цілісності, які підтримуються мовою більш високого рівня.

Теоретичні мови запитів

Реляційна алгебра — це формальна мова, яка складається з набору операцій над реляціями (таблицями), таких як об'єднання, перетин, різниця, селекція, проєкція, з'єднання та ділення.



Основні операції реляційної алгебри:

- **Селекція (σ)** — вибір рядків, які відповідають певному критерію.
- **Проєкція (π)** — вибір окремих стовпців з реляції.
- **Об'єднання (∪)** — злиття двох реляцій з унікальними рядками.
- **Різниця (−)** — видалення рядків однієї реляції з іншої.
- **З'єднання (⋈)** — комбінування рядків з двох реляцій на основі спільних атрибутів.

Теоретичні мови запитів

Реляційне числення — це декларативна мова запитів, яка дозволяє описувати запити за допомогою логічних формул. На відміну від реляційної алгебри, яка фокусується на тому, як виконувати запити, реляційне числення описує, що саме потрібно отримати.

Реляційне числення поділяється на два типи:

- **Числення кортежів (Tuple Relational Calculus, TRC)** — запити формулюються у вигляді логічних виразів над кортежами.
- **Числення доменів (Domain Relational Calculus, DRC)** — запити базуються на логічних виразах над окремими атрибутами.

Теоретичні мови запитів

Хоча реляційна алгебра та числення є основою для роботи з реляційними базами, існують також теоретичні підходи для роботи з нереляційними структурами даних, такими як XML, JSON або графи.

- **XQuery** — мова запитів для роботи з XML-документами, заснована на теорії деревовидних структур.
- **SPARQL** — мова запитів для роботи з RDF-графами, яка базується на формальній логіці та теорії графів.

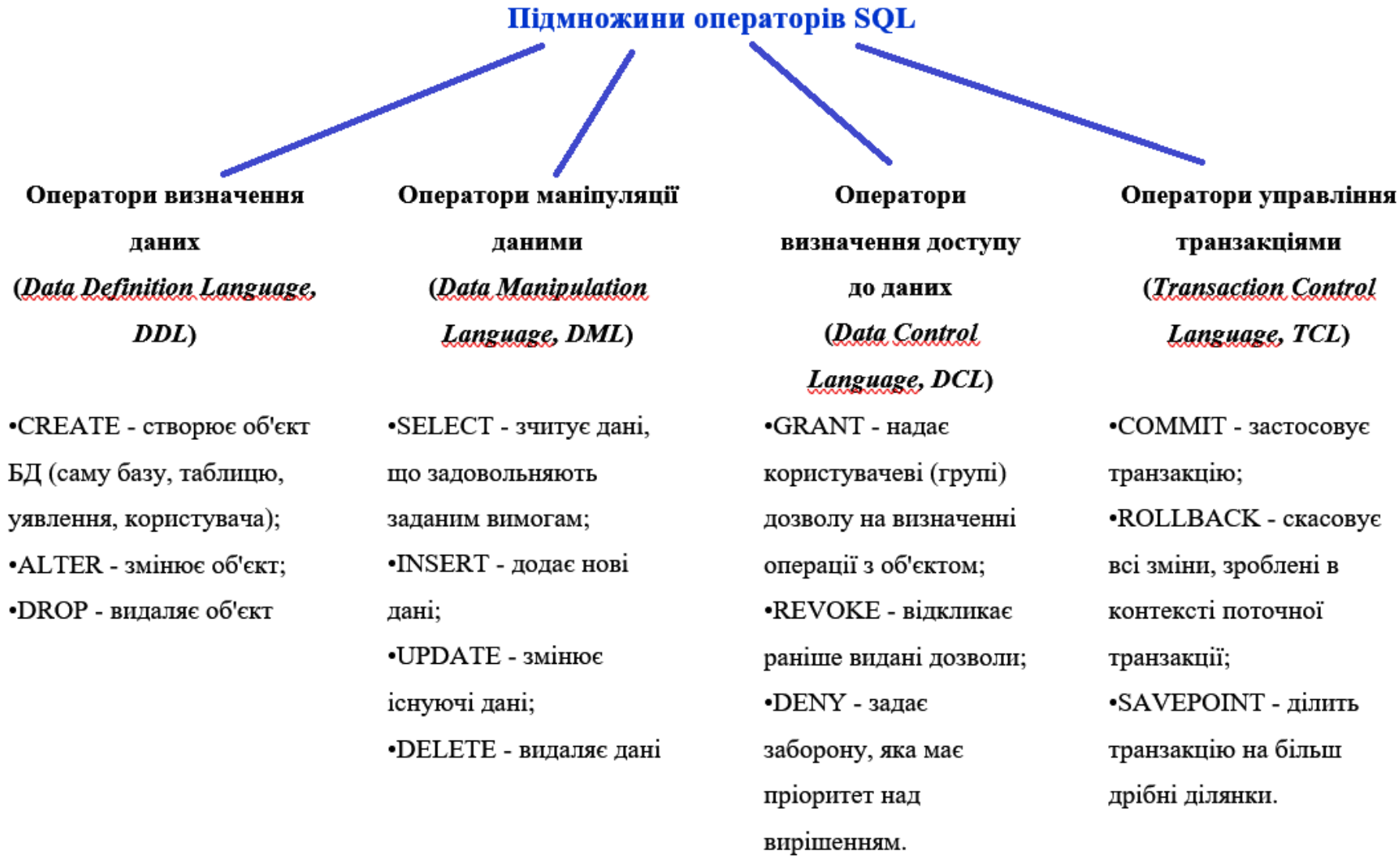
Мова запитів SQL

Запит є спеціальним чином описана вимога, що визначає склад вироблених над БД операцій по вибірці, віддаленню або модифікацій збережених даних.

Для підготовки запитів за допомогою різних СУБД найчастіше використаються дві основні мови:

QBE (Query By Example) - мова запитів за зразком;
SQL (Structured Query Language) - структурована мова запитів.

Мова запитів SQL



Оператори визначення даних (*Data Definition Language, DDL*)

- CREATE - створює об'єкт БД (саму базу, таблицю, уявлення, користувача);
- ALTER - змінює об'єкт;
- DROP - видаляє об'єкт

Оператори маніпуляції даними (*Data Manipulation Language, DML*)

- SELECT - зчитує дані, що задовольняють заданим вимогам;
- INSERT - додає нові дані;
- UPDATE - змінює існуючі дані;
- DELETE - видаляє дані

Метасимволи оператора LIKE

Стандартними символами-шаблонами є:

символ підкреслення (_), який можна застосовувати замість будь-якого одиничного символу в значенні що перевіряється;

символ відсотка (%) замінює послідовність будь-яких символів (число символів в послідовності може бути від 0 і більше);

[] - одиночний символ з набору символів (наприклад, [zxy]) або діапазону ([a-z]), зазначених у квадратних дужках. При цьому можна перечислити відразу кілька діапазонів (наприклад, [0-9a-z]);

^ - який в поєднанні з квадратними дужками виключає з пошукового зразка символи з набору або діапазону.

Операції об’єднання та групування

Отримання підсумкових значень

COUNT (*)	Повертає кількість рядків джерела записів
COUNT	Повертає кількість значень у вказаному стовпці
SUM	Повертає суму значень в зазначеному стовпці
AVG	Повертає середнє значення в зазначеному стовпці
MIN	Повертає мінімальне значення в зазначеному стовпці
MAX	Повертає максимальне значення в зазначеному стовпці

Об'єднання таблиць. Внутрішні об'єднання

Для об'єднання запитів використовується службове слово **UNION**:

<запит 1>
UNION [ALL]
<запит 2>

Зовнішнє об'єднання

Синтаксис для зовнішнього об’єднання наступний:

```
SELECT ім’я_таблиці_1.ім’я_стовбца, ім’я_таблиці_2.ім’я_стовбца
FROM ім’я_таблиці_1 ТИП ОБ'ЄДНАННЯ ім’я_таблиці_2
ON умова _ об'єднання;
```

де ТИП ОБ'ЄДНАННЯ може бути наступним:

INNER JOIN – є найпростіший тип об'єднання, де кожен рядок в одній таблиці зіставляється з усіма іншими рядками в іншій таблиці. Тільки якщо умова з'єднання оцінюється як істинне. Загальні дані повинні бути істинними для обох таблиць, що беруть участь в JOIN.

LEFT OUTER JOIN - вказує на те, що з таблиці зліва треба взяти все рядки і з'єднати з рядками з іншої таблиці.

RIGHT OUTER JOIN - вказує на те, що з таблиці праворуч треба взяти все рядки і з'єднати з рядками з іншої таблиці.

FULL OUTER JOIN - повне зовнішнє об'єднання, яке отримає всі рядки з обох таблиць і зв'яже між собою ті, які можуть бути пов'язані.

Групування записів

Для групування в SQL використовується оператор GROUP BY.

```
SELECT maker, COUNT(type) AS Kol
FROM Product
GROUP BY maker
HAVING COUNT (type)> 2
```

Вище ми розглядали, які умови можна задавати оператором *WHERE*, ті ж умови можна задавати і оператором *HAVING*, тільки треба запам'ятати, що *WHERE* фільтрує рядки, а *HAVING* - групи.

Концептуальна модель бази даних.

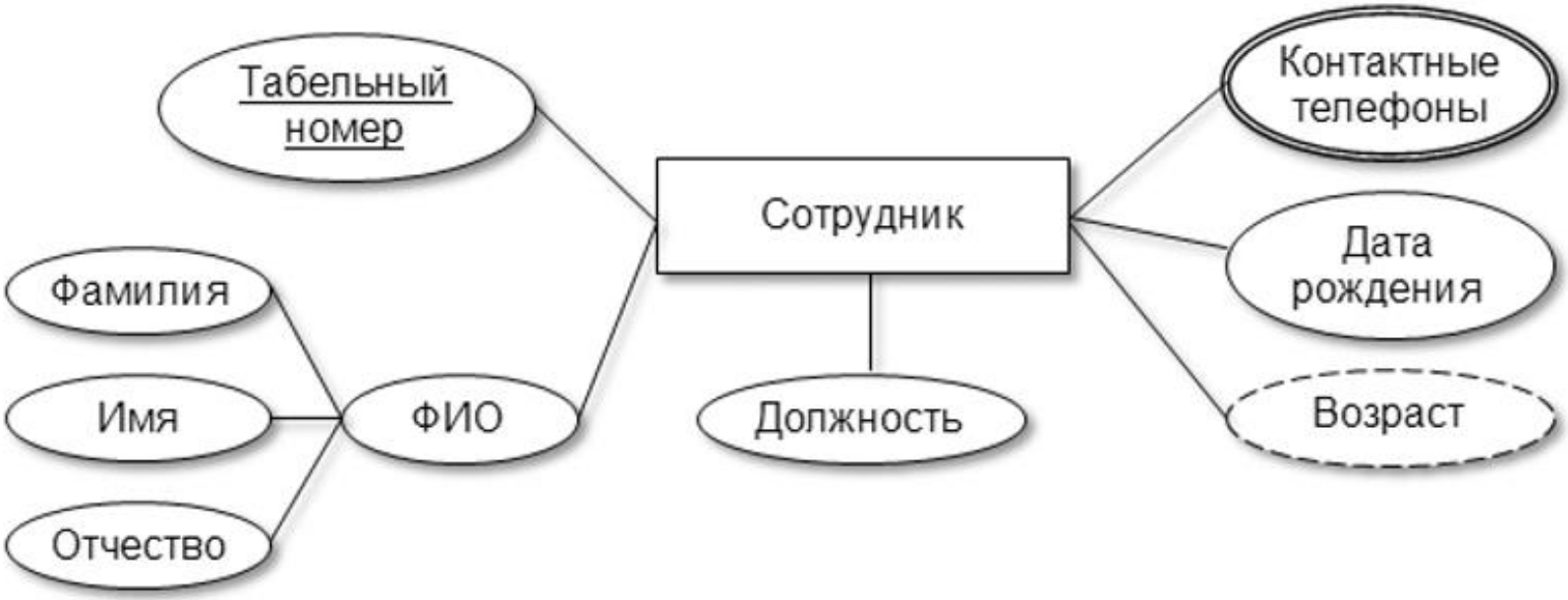
Концептуальна модель бази даних - це наочна діаграма, намальована в прийнятих позначеннях і яка докладно показує зв'язок між об'єктами та його характеристиками.

Діаграму сутність/відносини називають ER-діаграмою (Essence (сутність) та Relation (зв'язок)).

Позначення ER-діаграми за Пітером Ченом.

Запропонована Ченом графічна нотація ERD включає такі елементи.

Незалежна сутність відображається на діаграмах прямокутником із одинарною рамкою, залежна – з подвійною



Потужність (кардинальність) зв'язку позначається числами або літерами:

- 1 – один;
- N або M – багато;
- <число> - конкретна кількість екземплярів;
- (< Min >, < Max >) – діапазон екземплярів, де як < Min > і < Max > можуть використовуватися попередні позначення потужності.

а) односимвольне позначення

б) діапазонне позначення

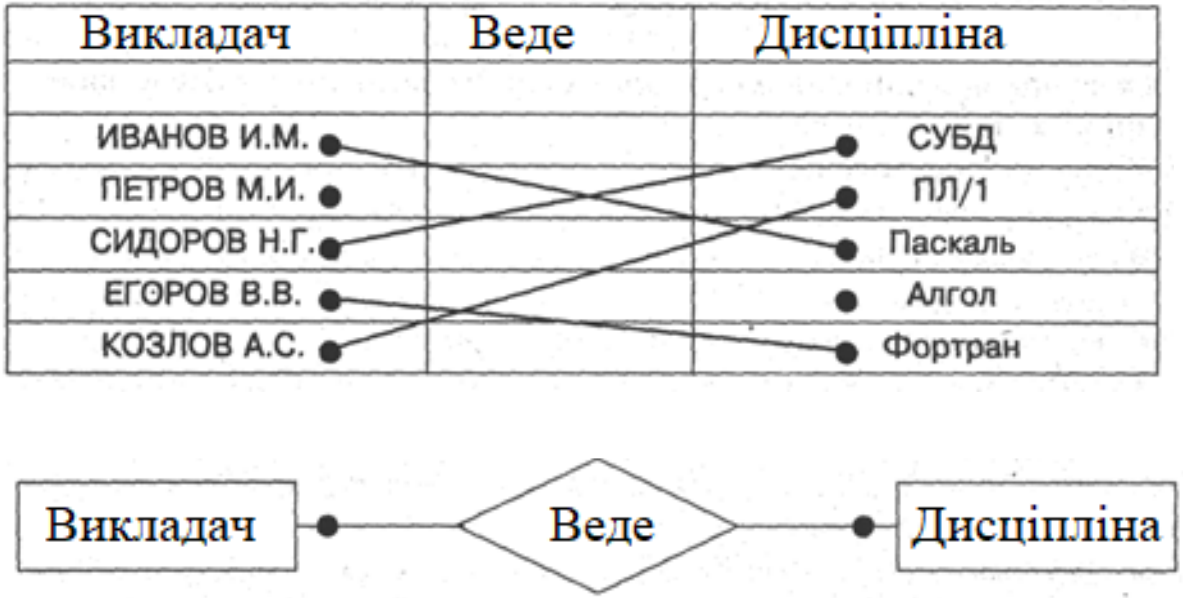
Нотація Gordon Everest

Концептуальна модель бази даних ERD Fork. Зв'язок сутностей Artist та Song з використанням нотації Crow's Foot. Пісня (Song) має «одного і лише одного» виконавця (Artist); виконавець пов'язаний з «нулем, однією чи кількома» піснями

Основні поняття методу ERD

Основними поняттями методу сутність-зв'язок є такі:

- суть,
- атрибут сутності,
- ключ сутності,
- зв'язок між сутностями,
- ступінь зв'язку,
- клас власності екземплярів сутності,
- діаграми ER-екземплярів,
- діаграми ER-типу.



Клас приналежності (КП) сутності може бути: обов'язковим та необов'язковим.

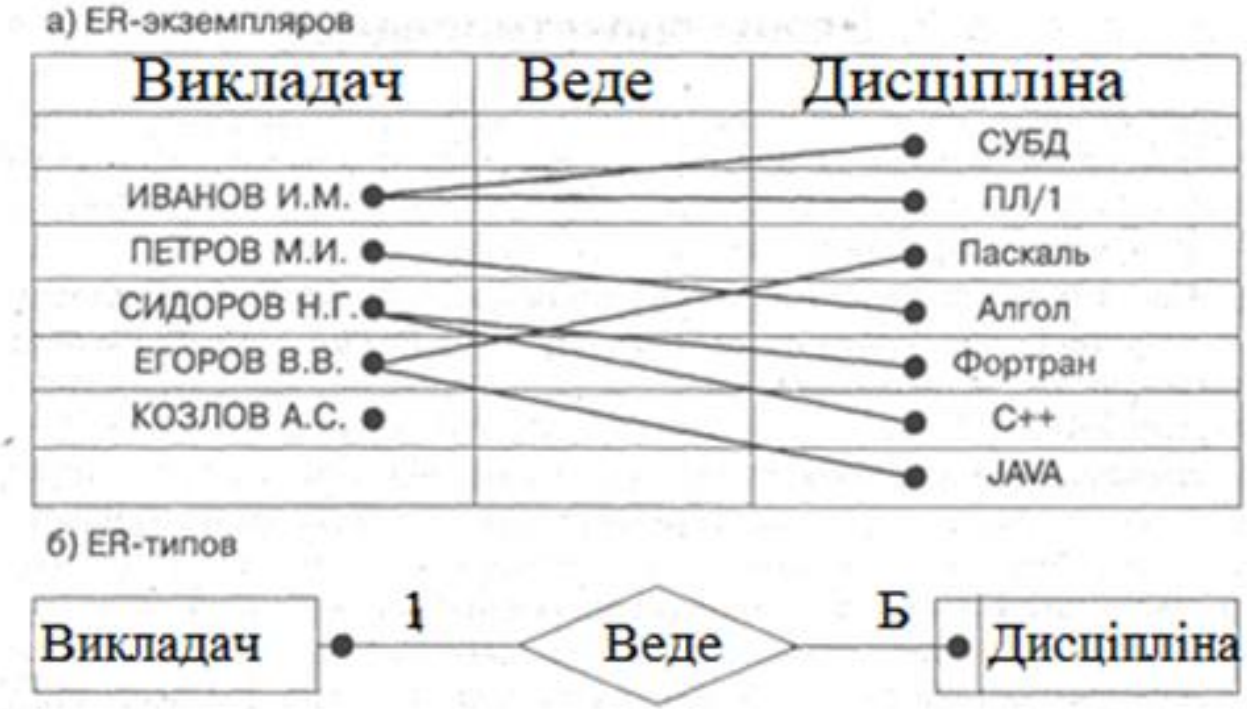
Клас приналежності сутності є обов'язковим, якщо всі екземпляри цієї сутності обов'язково беруть участь у аналізованому зв'язку, інакше клас належності сутності є необов'язковим.

а) ER-екземпляров

Викладач	Веде	Дисципліна
ИВАНОВ И.М.		СУБД
ПЕТРОВ М.И.		ПЛ/1
СИДОРОВ Н.Г.		Паскаль
ЕГОРОВ В.В.		Алгол
КОЗЛОВ А.С.		Фортран

б) ER-типов

Позначимо обов'язковий клас власності символом «О», а необов'язковий - символом «Н», тоді, варіанти зв'язку типу 1:Б умовно можна як: О-О, О-Н, Н-О, Н-Н



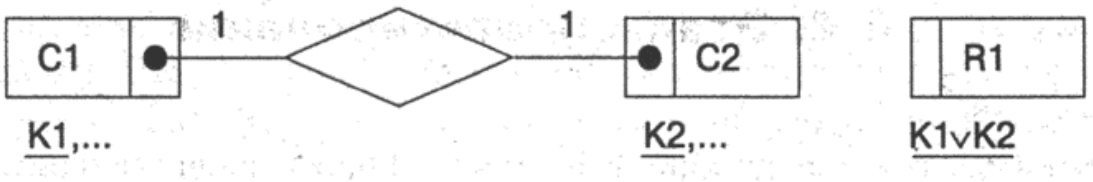
Етапи проектування концептуальної моделі

Процес проектування бази даних є ітераційним, що допускає повернення до попередніх етапів для перегляду раніше прийнятих рішень і включає наступні етапи:

1. Виділення сутностей та зв'язків між ними.
2. Побудова діаграм ER-типу з урахуванням усіх сутностей та їх зв'язків.
3. Формування набору попередніх відносин із зазначенням передбачуваного первинного ключа для кожного відношення та використанням діаграм ER-типу.
4. Додавання не ключових атрибутів у відносинах.
5. Приведення попередніх відносин до нормальної форми Бойса -Кодда, наприклад, за допомогою методу нормальних форм.
6. Перегляд ER-діаграм у таких випадках:
   - деякі відносини не наводяться до нормальної форми Бойса – Кодда;
   - деяким атрибутам немає логічна обґрунтованих місць у попередніх відносинах.

Формування відносин для зв'язку 1:1

Правило 1. Якщо ступінь бінарного зв'язку 1:1 та клас належності обох сутностей є обов'язковим, то формується одне відношення. Первинним ключем цього відношення може бути ключ будь-якої з двох сутностей.



На рисунку використовуються такі позначення:

C1, C2 - сутності 1 та 2;

Kl , K2 - ключі першої та другої сутності відповідно;

R1 - відношення 1, сформоване на основі першої та другої сутностей;

KlvK 2,... означає, що ключем сформованого відношення може бути або К1 або К2.

Викладач_Дисципліна

ІН	ПІБ	Стаж	КД	Год.
П1	Иванов И.М.	5	К1	62
П2	Петров М.И.	7	К2	74
П3	Сидоров Н.Г.	10	К3	102
			К4	80

Правило 2. Якщо ступінь зв'язку 1:1 та клас приналежності однієї сутності обов'язковий, а другий - необов'язковий, то під кожну з сутностей формуються первини ключі, які є ключами відповідних сутностей.

Схема та відношення, отримані за правилом 1 (початкове ставлення)

Викладач

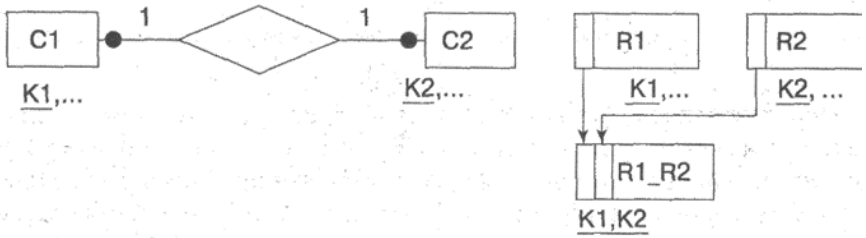
ІН	ПІБ	Стаж	КД
П1	Иванов И.М.	5	К1
П2	Петров М.И.	7	К2
П3	Сидоров Н.Г.	10	К3
П4	Егоров В.В.	5	К4

Дисципліна

КД	Год
К1	62
К2	74
К3	102
К4	80

Відносини, отримані за правилом 2

Правило 3 Якщо ступінь зв'язку 1:1 та клас приналежності обох сутностей є необов'язковим, необхідно використовувати три відносини. Два відносини відповідають сутностям, що зв'язуються, ключі яких є первинними в цих відносинах. Третє відношення є зв'язковим між першими двома, тому його ключ поєднує ключові атрибути зв'язуваних відносин.



ІН	ПІБ	Стаж
П1	Иванов И.М.	5
П2	Петров М.И.	7
П3	Сидоров Н.Г.	10

ІН	КД
П1	К1
П3	К2

КД	Год
К1	62
К2	74
К3	102

Варіанти відносин для правила 3

Формування відносин для зв'язку 1:Б

Викладач_дисципліна

ІН	ПІБ	Стаж	КД	Год
П1	ІвановИ.М.	5	К1	62
П1	ІвановИ.М.	5	К2	74
П2	ПетровМ.И.	7	К4	80
П3	Сидоров Н.Г.	10	К5	96
П3	Сидоров Н.Г.	10	К6	120
П4	Егоров В.В.	5	К3	102
П4	Егоров В.В.	5	К7	89
П5	Козлов А.С.	8	---	---

Початкове ставлення

Правило 4. Якщо ступінь зв'язку між сутностями 1:Б (або Б:1) і клас приналежності Б-зв'язної сутності обов'язковий, достатньо формування двох відносин (по одному на кожну з сутностей). При цьому первинними ключами цих відносин є ключі їх сутностей.

Викладач

ІН	ПІБ	Стаж
П1	ІвановИ.М.	5
П2	ПетровМ.И.	7
П3	Сидоров Н.Г.	10
П4	Егоров В.В.	5
П5	Козлов А.С.	8

Дисципліна

КД	Год	ІН
К1	62	П1
К2	74	П1
К3	102	П4
К4	80	П2
К5	96	П3
К6	120	П3
К7	89	П4

Відносини, отримані за правилом 4

Правило 5. Якщо ступінь зв'язку 1:Б (Б:1) та клас приналежності Б-зв'язної сутності є необов'язковим, необхідно формування трьох відносин. Два відносини відповідають сутностям, що зв'язуються, ключі яких є первинними в цих відносинах. Третє відношення є зв'язковим між першими двома (його ключ поєднує ключові атрибути зв'язуваних відносин).

Викладач_Дисципліна

ІН	ПІБ	Стаж	КД	Год
П1	ИвановИ.М.	5	К1	62
П1	ИвановИ.М.	5	К2	74
П2	ПетровМ.И.	7	К4	80
---	---	---	К5	96
П3	Сидоров Н.Г.	10	К6	120
П4	Егоров В.В.	5	К3	102
П4	Егоров В.В.	5	К7	89
П5	Козлов А.С.	8	---	---

Вихідне ставлення

Викладач

ІН	ПІБ	Стаж
П1	ИвановИ.М.	5
П2	ПетровМ.И.	7
П3	Сидоров Н.Г.	10
П4	Егоров В.В.	5
П5	Козлов А.С.	8

Веде

ІН	КД
П1	К1
П1	К2
П2	К4
П3	К6
П4	К3
П4	К7

Дисципліна

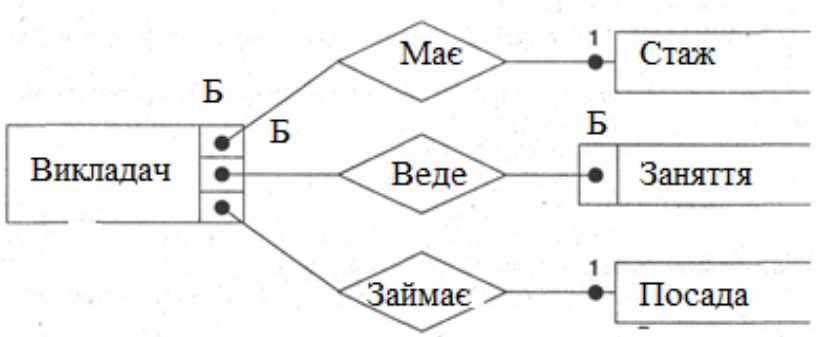
КД	Год
К1	62
К2	74
К3	102
К4	80
К5	96
К6	120
К7	89

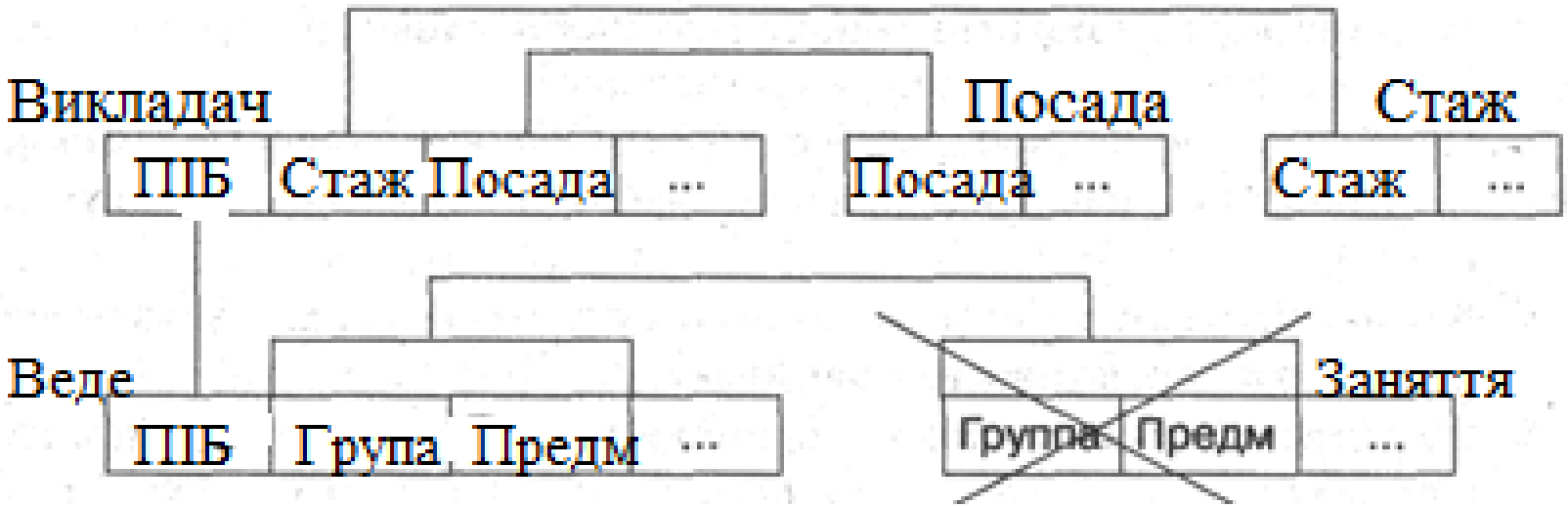
Приклад проектування концептуальної моделі навчальної частини

Перший етап проектування – виділення сутностей та зв'язків між ними.

Виділимо такі сутності:	Виділимо зв'язки між сутностями:
ВИКЛАДАЧ (Ключ - ПІБ).	Викладач має СТАЖ,
ЗАНЯТТЯ (Ключ – Група, Предм)	Викладач веде заняття,
СТАЖ (Ключ – Стаж)	Викладач займає посаду.
ПОСАДА (Ключ – Повинний)	

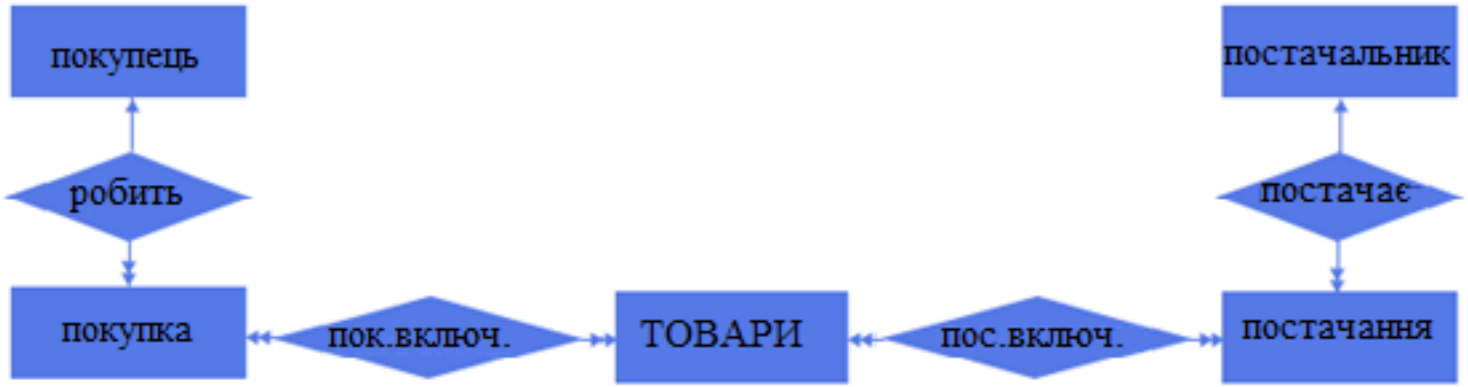
Другий етап проектування - побудова діаграми ER-типу з урахуванням усіх сутностей та зв'язків між ними.



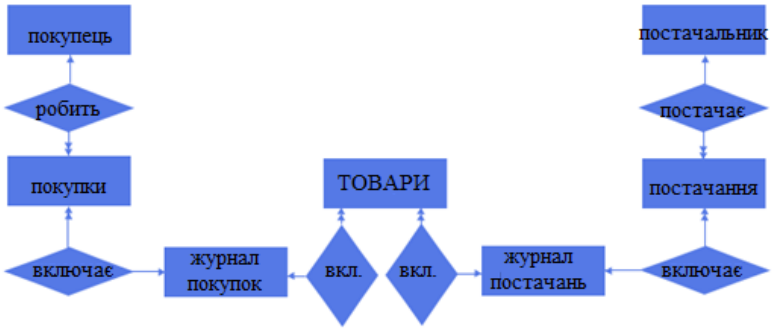

Викладач
ПІБ
Стаж
Посада
...
Посада
Посада
...
Стаж
Стаж
...
Веде
ПІБ
Група
Предм
...
Заняття
Група
Предм
...

Приклад проектування концептуальної моделі інтернет-магазину

Об'єкти та зв'язки (1-е наближення)



Об'єкти та зв'язки (2-е наближення)



Додавання проміжного об'єкта

Атрибути та об'єкти інтернет-магазину

Покупатель
Id покупателя (PK)
ФИО
E-mail

Поставщик
Id поставщика (PK)
Наименование
Адрес

Покупка
Id покупки (PK)
Id покупателя (FK)
Дата

Поставка
Id поставки (PK)
Id поставщика (FK)
Дата

Товар
Id товара (PK)
Наименование
Цена

Журнал покупок	
Id покупки (FK)	} PK
Id товара (FK)	
Количество	

Журнал поставок	
Id поставки (FK)	} PK
Id товара (FK)	
Количество	

Набір попередніх таблиць

Поставщик
Id поставщика (PK)
Наименование
Город
Адрес

Покупатель
Id покупателя (PK)
ФИО
E-mail

Поставщик
Id поставщика (PK)
Наименование
Адрес

Товар
Id товара (PK)
Наименование
Цена

Журнал покупок	
Id покупателя (FK)	PK
Дата	
Id товара (FK)	
Количество	

Журнал поставок	
Id поставщика (FK)	PK
Дата	
Id товара (FK)	
Количество	

Товар
Id товара (PK)
Наименование
Дата
Цена

Товар
Id товара (PK)
Наименование

Цена	
Id товара (FK)	PK
Дата	
Цена	


Покупатель
Id покупателя (PK)
ФИО
E-mail
Поставщик
Id поставщика (PK)
Наименование
Город
Адрес
Покупка
Id покупки (PK)
Id покупателя (FK)
Дата
Поставка
Id поставки (PK)
Id поставщика (FK)
Дата
Журнал покупок
Id покупки (FK)
Id товара (FK)
Количество
PK
Журнал поставок
Id поставки (FK)
Id товара (FK)
Количество
PK
Товар
Id товара (PK)
Наименование
Цена
Id товара (FK)
Дата
Цена
PK