МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА
ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ

Кафедра цифрової економіки та бізнес-аналітики

КУРСОВА РОБОТА
з навчальної дисципліни
“Проектування та адміністрування БД і СД”

Тема:

«Інформаційна система для фірми, що виробляє електротехніку»

Науковий керівник:
к.е.н., доц. Депутат Б.Я.
(прізвище, ім’я, по-батькові)
_____ (підпис)

“___” _____ 2020 р.

Виконавець:
Саламаха О.Ю.
(прізвище, ім’я, по-батькові)
УФЕ-31с група
_____ (підпис)

“___” _____ 2020 р.

Загальна кількість балів _____
(підписи, ПІП членів комісії)

Львів 2020

ЗМІСТ

ВСТУП

Неможливо уявити сучасний світ без електроприладів, вони стали настільки звичними для людей що ми просто не можемо від них відмовитися. Для того щоб користуватися електроприладами будь-яких типів їх спочатку потрібно виготовити та реалізувати.

Фірмі що виробляє електротехніку потрібна інформаційна система, щоб організувати виробництво та закупку такої техніки. Бази даних, що створюються є схожими на картотеки, але на відміну від записної книжки можуть містити сотні і тисячі записів, що зберігають сукупність взаємопов'язаної інформації про ті чи інші об'єкти. Головна перевага, яку нам дає перехід до автоматизованого ведення бази даних – швидкий пошук необхідної інформації і представлення її в потрібній формі. Використання баз даних є однією з характерних рис більшості сучасних інформаційних систем. По своїй суті бази даних є тим, навколо чого і будується інформаційна система будь-якого підприємства. Сучасні бази даних повинні забезпечувати багатомашинну обробку та зберігання великих обсягів інформації, оперативний аналіз даних.

Обрана тема курсової роботи актуальною тому, що інформаційна система забезпечує функціонування підприємства.

Метою курсової роботи є дослідження теоретичних основ ІС та її реалізація.

Завданнями курсової роботи є:

- аналіз предметної області та створення БД інструментами SQL.
- забезпечити редагування таблиць БД та взаємозв'язків між різними таблицями.

Об’єктом дослідження курсової роботи є процес розробки інформаційної системи для фірми що виробляє електротехніку з використанням системи управління базами даних SQL.

Предметом дослідження є теоритичні основни та практичне застосування технологій SQL, mySQL workbench.

Інформаційною базою для написання курсової роботи по даній темі стали, підручники і навчальні посібники, інформаційні матеріали з Інтернету.

Робота скадається з двох розділів. У першому розділі описано бізнес-ідею та аналіз вимог. У наступному описано побудову навчальної бази даних.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА АНАЛІЗ РОБОТИ ФІРМИ ЩО ВИРОБЛЯЄ ЕЛЕКТРОТЕХНІКУ.

1.1 Дослідження роботи фірми що виготовляє електротехніку.

Тема досліджуваної предметної області - «Фірма що виробляє електротехніка». Фірма виготовляє електротехніку. Сучасні технології дозволяють вирішувати будь-які поставлені завдання. Вони пропонують інструменти для ведення бізнесу в Інтернеті, що впливають на прибуток підприємства, його популярність і попит на його продукцію (послуги).

Головною метою фірми є розробка та реалізація електротехніки з максимальною автоматизацією виробництва та ефективістю.

1.2 Постановка завдання.

Досягнення поставленої мети можливо за допомою автоматизації процесу виробництва тобто створення бази даних яка містила б у собі всю інформацію про товари та послуги фірми.

База даних повинна бути максимально інформативною та економити час на виробництво товарів та послуг. Для цього у базі буде така інформація:

1. Інформація про офіси (назва, тип, телефон, адреса).
2. Інформація про працівників (ім’я, прізвище, посада, офіс).
3. Інформація про посади (назва , зарплата, бонуси, офіс).
4. Інформація про товари (назва, тип, категорія, ціна).
5. Інформація про покупця (ім’я, тип).
6. Інформація про замовлення (тип, товар, офіс, отримувач, кількість).
7. Інформація про рахунок (тип, товар, офіс, працівник, покупець, кількість, сума).

РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ

2.1 Основні поняття концептуального проектування

Концептуальне (інфологічне) проєктування - побудова семантичної моделі предметної області, тобто інформаційної моделі найбільш високого рівня абстракції. Така модель створюється без орієнтації на якусь конкретну СУБД і модель даних. Терміни «семантична модель», «концептуальна модель» і «інфологіческая модель» є синонімами.

Конкретний вид і зміст концептуальної моделі бази даних визначається обраним для цього формальним апаратом. Зазвичай використовуються графічні нотації, подібні ER-діаграм.

Найчастіше концептуальна модель бази даних включає в себе:

-опис інформаційних об'єктів або понять предметної області та зв'язків між ними.

-опис обмежень цілісності, тобто вимог до допустимих значень даних і до зв'язків між ними.

Логічне проектування - створення схеми бази даних на основі конкретної моделі даних, наприклад, реляційної моделі БД. Для реляційної моделі даних - це набір схем відносин, зазвичай із зазначенням первинних ключів, а також «зв'язків» між відносинами, що представляють собою зовнішні ключі.

Перетворення концептуальної моделі в логічну модель, як правило, здійснюється за формальними правилами. Цей етап може бути в значній мірі автоматизований.

На етапі логічного проєктування враховується специфіка конкретної моделі даних, але може не враховуватися специфіка конкретної СУБД.

Фізичне проєктування - створення схеми бази даних для конкретної СУБД. Специфіка конкретної СУБД може включати в себе обмеження на іменування об'єктів бази даних, обмеження на підтримувані типи даних та інші. Крім того, специфіка конкретної СУБД при фізичному проєктуванні включає вибір рішень, пов'язаних з фізичним середовищем зберігання даних (вибір методів управління дисковою пам'яттю, поділ БД по файлам і пристроям, методів доступу до даних), створення індексів та інші.

Види зв’язку між двома об’єктами бувають:

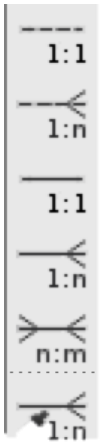
1:1 -Один до одного: Обидві таблиці можуть мати лише один запис з обох сторін відносин. Кожне значення первинного ключа стосується лише одного запису у відповідній таблиці. Вони схожі на пари - ви можете або не можете бути парами.

1:n - Один до багатьох: Таблиця первинного ключа містить лише один запис, що стосується однієї чи багатьох записів у відповідній таблиці. Ці стосунки схожі на стосунки між вами та матір’ю. У вас є лише одна мати, але у матері може бути кілька дітей.

n:m - Багато-до багатьох: кожен запис в обох таблицях може стосуватися будь-якої кількості записів (або відсутніх записів) в іншій таблиці. Відносини багатьох до багатьох потребують третьої таблиці, відомої як

асоційована або пов’язуюча таблиця, оскільки реляційні системи не можуть безпосередньо вмістити відносини.

Рис.1



2.2 Розробка концептуальної моделі даних (ER-діаграми).

Концептуальна модель бази даних - модель, яка визначає систему основних понять і правил їх комбінування, які не залежать від засобів розробки бути смислового структурою предметної області. Для представлення концептуальної моделі бази даних створюється діаграма «сутність-зв'язок» (ERD). Основними конструктивними елементами є сутності, зв'язки між ними та їх властивості (атрибути).

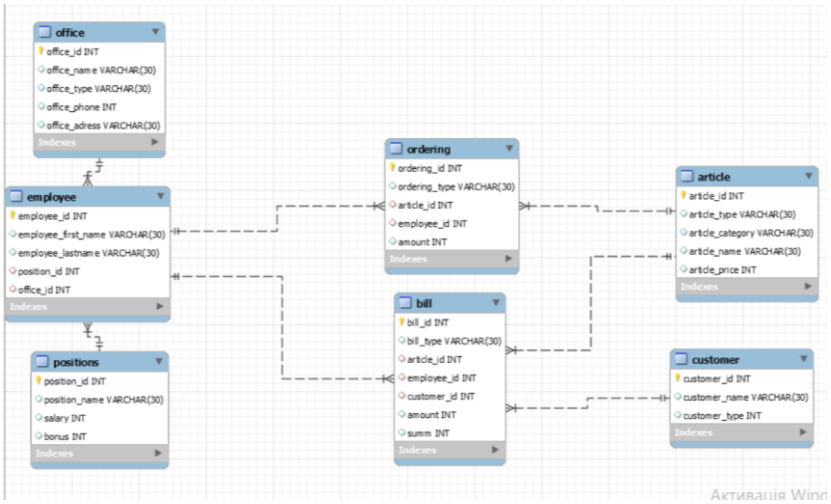
Сутність - будь-який чудовий об'єкт. Сутність володіє одним або декількома атрибутами, які або належать суті, або успадковуються через зв'язок. У даній роботі сутностями є: «Studio», «Services», «Workers», «Contact».

Останнім кроком моделювання є ідентифікація атрибутів. Атрибут - проіменована характеристика сутності. Всі атрибути позначаються через овал. Екземпляр атрибута - це певна характеристика окремого елемента множини. Екземпляр атрибута визначається типом характеристики і її значенням, названим

значенням атрибута. В ER-моделі атрибути асоціюються не тільки з конкретними сутностями, але й зв'язками. Таким чином, екземпляр сутності повинен мати єдине певне значення для асоційованого атрибута. Атрибути використовуються для визначення того, яка інформація повинна бути зібрана про сутності.

Складемо діаграму сутностей – ER-діаграму для фірми що виробляє електротехніку:

Рис.2



2.2.1 Перелік таблиць бази даних

База даних містить такі таблиці :

1. Office – містить усі дані про офіси компанії

2. Employee – містить усі дані про працівників компанії

3. Positions – мстить усі дані про посади в компанії

4. Article – список товарів, що виробляється

5. Ordering – демонструє рух товарів всередині компанії

6. Bill – рахунки

7. Customer – покупці

2.2.2 Перелік полів таблиць бази даних

Поля які ідентифікують властивості “office”

office_id – айді офісу

office_name – імя офісу

office_type – тип офісу

office_phone – контактний телефон

office_adress – адреса

Поля які ідентифікують властивості “employee”

employee_id – айді працівника

employee_first_name – імя працівника

employee_lastname – прізвище працівника

position_id – айді позиції

office_id – айд офісу

Поля які ідентифікують властивості “positions”

position_id –айді посади

position_name – назва посади

salary – заробітна плата

bonus - бонуси

office_id – айфді офісу

Поля які ідентифікують властивості “article”

article_id – айді товару

article_type – тип товару

article_category – категорія товару

article_name – назва товару

article_price – ціна на одиницю товару

Поля які ідентифікують властивості “ordering”

ordering_id – айді замовлення

ordering_type – тип замовлення

article_id – тип товару

office_id – айді офісу

employee_id – айді працівника

amount – кількість товару

Поля які ідентифікують властивості “Bill”

bill_id – айді рахунку

bill_type – тип рахунку

article_id – айді товару

office_id – айді офісу

employee_id – працівник який виконав операцію

customer_id – айді покупця

amount – кількість товару

summ –сума покупки

Поля які ідентифікують властивості “customer”

customer_id – айді покупця

customer_name – ім’я покупця

customer_type – тип покупця

2.3 Нормалізація реляційних відношень.

Нормалізація - це розбивка таблиці на дві або більше частин, які характеризуються кращими властивостями при доповненні, зміні і вилученні даних. Кінцева мета нормалізації зводиться до отримання такого проекту бази даних, у котрому кожний факт з'являється лише в однім місці, тобто виключена надлишковість інформації. Це робиться не стільки з метою економії пам'яті, скільки для виключення можливої суперечливості збережених даних.

Перша нормальна форма (1НФ) - це звичайне відношення, на перетині рядків і стовпів якого розташоване атомарне значення відповідного атрибута, що має унікальне значення ключового атрибута, що не допускає дублювання кортежів. Таке відношення автоматично вже знаходиться в 1НФ.

Відношення знаходиться в другій нормальній формі (2НФ) тоді,коли відношення знаходиться в 1НФ і немає неключових атрибутів, залежних від частини складного ключа. Неключовий атрибут - це атрибут, який не входить до складу жодного потенційного ключа. Варто зазначити, що якщо потенційний ключ відношення є простим, то відношення автоматично перебуває під 2НФ.

Відношення знаходиться в третій нормальній формі (3НФ) тоді, коли відношення знаходиться в 2НФ і всі неключові атрибути взаємно незалежні.

Атрибути називаються взаємно незалежними, якщо жоден з них не є функціонально залежним від іншого.

Поняття функціональної залежності є базовим, так як на його основі формулюються визначення всіх інших видів залежностей. Функціональна залежність - це зв'язок між атрибутами.

У відношенні R атрибут Y функціонально залежить від атрибута X (X і Y можуть бути складовими) у тому випадку, якщо кожному значенню X відповідає в точності одне значення Y.

Функціональні залежності між атрибутами в таблиці “office”, котра містить усю інформацію про студію де ключовим полем є поле ідентифікаційний код офісу, тобто поле “office_id”, із значенням “Primary”. Поля “name”, “address”, “type”, ”phone”, залежні від поля “office_id”.

Далі розглянемо функціональні залежності між атрибутами в таблиці ”employee’ , у якій міститься інформація про послуги. Поля “firstname”,”lastname” залежні від поля “employee_id”, натомість поля “position_id” і “office_id” залежні від відповідних полів у інших таблицях, такі поля є вторинними ключами.

У таблиці “positions” усі поля залежні від “positions_id” окрім поля “ office_id”.

У таблиці “article” всі поля унікальні залежні від поля “article_id”.

У таблиці “bill” головне поле ”bill_id” залежні поля “summ, customer_id, amount, bill_type”, поля вторинними із ключами “article_id, office_id, employee_id”.

У таблиці “customer” всі поля унікальні та залежні від поля “customer_id”

У таблиці “order” поля залежні від поля “order_id”, “order_type,amount” інші це вторинні ключі у відповідних таблицях “article_id, office_id, employee_id”

2.4 Визначення типів даних.

Структура «SQL» в своїй структурі містить такі елементи:

-база даних;

-таблиці;

-записи таблиць.

employee_id INT AUTO_INCREMENT PRIMARY KEY, - первинний ключ

employee_first_name VARCHAR(30), - обов’язкове поле розмір 30 символів

employee_lastname VARCHAR(30), - обов’язкове поле розмір 30 символів

position_id INT, - вторинниий ключ на 10 символів

constraint fk_position_employee foreign key (position_id) REFERENCES positions (position_id), - посилання на вторинний ключ

office_id INT, вторинниий ключ на 10 символів

constraint fk_office_employee FOREIGN KEY (office_id) REFERENCES office (office_id)); - посилання на вторинний ключ

Першим етапом є створення бази даних “mySQL” із назвою «f1rm». Наступний етап полягає у наповненні бази даними які використовуються для функціонування сайту. Завершальним етапом є наповнення таблиць даними.

2.5 Обмеження цілісності даних.

Обмеження цілісності — це правила, які обмежують усі можливі стани бази даних, а також переходи з одного стану в інший. Таким чином, обмеження цілісності визначають множину «допустимих» станів і переходів між ними. База

даних перебуває в цілісному стані, якщо вона відповідає всім визначеним для неї вимогам цілісності.

У навчальній базі “f1rm” застосовано наступні обмеження щодо первинних ключів:

- primary key;
- increment;

2.6 Створення бази даних.

```
CREATE DATABASE f1rm; //створюємо бд
USE f1rm; //використовуєм бд
CREATE TABLE office( //створюєм таблицю
office_id INT AUTO_INCREMENT PRIMARY KEY, //прописуємо назви стовпців, задаємо їх типи даних
office_name VARCHAR(30),
office_type VARCHAR(30),
office_phone INT,
office_adress VARCHAR(30));
//заповнюємо таблицю
INSERT INTO office VALUES(NULL, 'salash', 'administration', 0997658120, 'Lviv,Kruvchuci 11');
INSERT INTO office VALUES(NULL, 'niisan', 'workshop', 880055335, 'Lviv,Zelena 5');
INSERT INTO office VALUES(NULL, 'gazda', ' warehouse ', 0685748413, 'Lviv,Kulparkivska 34');
INSERT INTO office VALUES(NULL, 'gospodar', 'warehouse', 0674565578, 'Lviv,Patona 23a');

USE f1rm;
CREATE TABLE employee(
```

```
employee_id INT AUTO_INCREMENT PRIMARY KEY,
employee_first_name VARCHAR(30),
employee_lastname VARCHAR(30),
position_id INT,
constraint fk_position_employee foreign key (position_id) REFERENCES positions (position_id), //створюємо і підтягуємо вторинні ключі
office_id INT,
constraint fk_office_employee FOREIGN KEY (office_id) REFERENCES office (office_id));
    insert into employee values(null,'Igor','Smirnof',1,1);

USE f1rm;
CREATE TABLE positions(
position_id INT AUTO_INCREMENT PRIMARY KEY,
position_name VARCHAR(30),
salary INT,
bonus int,
office_id int,
constraint fk_office_id foreign key (office_id) references office (office_id));
insert into positions Values(null, 'cleaner',7000,1000,1);

use f1rm;
create table article(
article_id INT AUTO_INCREMENT PRIMARY KEY,
article_type VARCHAR(30),
article_category VARCHAR(5),
article_name VARCHAR(30),
article_price INT);
```

```
INSERT INTO article VALUES(NULL, 'completed product', 'cellphone', 'iphone, 10', 50000 );
INSERT INTO article VALUES(NULL, 'completed product', 'cellphone', 'samsung, galaxy,9', 30000 );
INSERT INTO article VALUES(NULL, 'completed product', 'laptop', 'acer, aspire, e15', 22000 );
INSERT INTO article VALUES(NULL, 'part','vedeocard', 'nvidia, geforce, 1070ti', 8000 );
INSERT INTO article VALUES(NULL, 'completed product', 'laptop', 'lenovo, yoga', 12000 );
INSERT INTO article VALUES(NULL, 'completed product', 'tablet', 'nexus,7', 15000 );
INSERT INTO article VALUES(NULL, 'completed product', 'cellphone', 'samsung, galaxy, note', 18000 );
INSERT INTO article VALUES(NULL, 'completed product', 'tablet', 'hp, transformer', 25000 );
INSERT INTO article VALUES(NULL, 'part', 'procesor', 'intelcore, i9', 14000 );
INSERT INTO article VALUES(NULL, 'part', 'videocard', 'radeon, hd, graphics', 8000 );


use f1rm;
CREATE TABLE customer (
customer_id INT AUTO_INCREMENT PRIMARY KEY,
 customer_name VARCHAR(30),
  customer_type INT);
  insert into customer values(null,'eldorado',0);
  insert into customer values(null,'rozetka',1);
  insert into customer values(null,'electrosvit',2);
```

```
use f1rm;
CREATE TABLE ordering (
ordering_id INT AUTO_INCREMENT PRIMARY KEY,
ordering_type VARCHAR(30),
article_id int,
    constraint fk_article_id2 foreign key (article_id) references article (article_id),
    office_id int,
    constraint fk_office_id2 foreign key (office_id) references office (office_id),
    employee_id int,
    constraint fk_employee_id2 foreign key (employee_id) references employee (employee_id),
    amount int);
    insert into ordering values(null,'output',5,2,2,10);
    insert into ordering values(null,'input',4,3,2,5);


  use f1rm;
      Create table bill(
     bill_id INT AUTO_INCREMENT PRIMARY KEY,
     bill_type varchar(30),
        article_id int,
                constraint fk_article_id3 foreign key (article_id) references article (article_id),
        office_id int,
        constraint fk_office_id3 foreign key (office_id) references office (office_id),
        employee_id int,
```

```
                    constraint fk_employee_id3 foreign key (employee_id) references
employee (employee_id),
        customer_id int,
                        amount int,
          summ int);
          insert into bill values(null,'input',9,3,2,2,10,45000);
```

Створення запитів до бази даних

```
#показати таблицю article
select*from article;
#показати посади із зарплатою > 12000
select * from positions where salary > 12000;
#показати ім'я працівника на посаді директора
select  employee_first_name from employee where position_id ='director' ;
#показати перших двох працівників по алфавіту
select * from  employee order by employee_first_name asc limit 2;
#показати офіси номер телефону яких закінчужться на 8
select *from office where office_phone like '%8';
#показати тип товару готова продукція
select article_name, article_type from article
where article_type like 'completed product';
#показати суму цін всіх товарів по 1шт
select  sum(article_price) from article;
#показати посаду із бонусами між 500 і 1500
select position_name, bonus from positions where bonus between 500 and 1500;
#показати посаду із максимальною зарплатою
select position_name, Max(salary) from positions;
#показти 5 посад із найменшою зарплатою
select * from positions
group by position_name
```

```
order by min(Salary)
limit 5;
#показати товар з найменшою ціною
select * from article
group by article_name
order by min(article_price) ;
#показати тип товару запчастина
SELECT * from article
where article_type like 'part';
#показати обєднані 2 таблиці
SELECT*FROM employee join office on office.office_id=employee.office_id;
#показати і порахувати суму продажу товару
select article.article_name, article.article_price, bill.amount, bill.amount*article.article_price as sum from bill join article on bill.article_id=article.article_id
where article.article_price > 10000;
#підвищити зарплату
update positions set salary=salary*1.1 where position_id=4;
```

ВИСНОВКИ.

Під час розробки проекту бази даних для фірми, що виробляє електротехніку були проведені дослідження і детальний аналіз предметної області, були створені функціональні, концептуальна, логічна і фізична моделі бази даних. Також були детально проаналізовані всі елементи, з яких складається даний програмний продукт і залежності між ними. Для розробки даного програмного продукту було використано середовище розробки бази даних mySQL workbench.

Електронна база, розроблена в даній курсовій роботі, набагато підвищує ефективність роботи в процесі обліку надання послуг та роботи фірми ,що виробляє електротехніку, оскільки дозволяє швидко організувати пошуки необхідних послуг за їх назвами або коду, прискорився процес складання звітності. Набагато простіше й швидше Можна редагувати всю інформацію завдяки зручному інтерфейсу.

Дану базу даних можливо використовувати не лише в даній фірмі, але й інших що мають , які мають споріднену предметну область – тобто, які надають певні послуги інтернет ресурсів або будь-яких інших.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

1. https://gist.github.com/AgentO3/670418d51a9103aeec55a0ec6a6b22ba

2. https://stackoverflow.com

3. 11 типів сучасних баз даних: короткий опис, схеми і приклади БД https://senior.ua

4. http://citforum.ru/database/dblearn

5. Внешние ключи FOREIGN KEY https://metanit.com/sql/mysql

6. Зовнішні ключі https://www.apserver.org.ua

7. Relational databases: Defining relationships between database tables https://www.techrepublic.com