Київ-2021 рік

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»**

| Факультет | Інформатики та обчислювальної техніки |
| --- | --- |
|  | (повна назва) |
| Кафедра | Інформатики та програмної інженерії |
|  | (повна назва) |
| Дисципліна | Бази даних |

Курс *2* Група *ІП-02*  Семестр *1*

**ЗАВДАННЯ**

**НА КУРСОВУ РОБОТУ СТУДЕНТУ**

| *Полішко Андрію Олексійовичу* |
| --- |

(прізвище, ім’я, по батькові)

| 1. Тема роботи | *База даних для підтримки діяльності рекламного агентства* |
| --- | --- |
| *сесії* | |
|  | |
|  | |
| керівник роботи | *старший викладач Марченко Олена Іванівна* |

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

| 1. Строк подання студентом роботи | | | *24.12.2021* |
| --- | --- | --- | --- |
| 1. Вихідні дані до роботи | |  | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
| 1. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно | | | |
| розробити) | *Вступ, аналіз предметної області, постановка завдання,* | | |
| *концептуальна модель бази даних, логічна модель бази даних, реалізація бази* | | | |
| *даних, висновки* | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
| 1. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) | | | |
|  | | | |
|  | | | |
|  | | | |
| 1. Дата видачі завдання | *31.10.2021* |

| № з/п | Назва етапів виконання курсового проекту | Строк виконання етапів проекту | Примітка |
| --- | --- | --- | --- |
| 1 | Вивчення літератури | *05.11.2021* |  |
| 2 | Аналіз предметного середовища | *10.11.2021* |  |
| 3 | Побудова ER-діаграми | *12.11.2021* |  |
| 4 | Побудова реляційної схеми | *22.11.2021* |  |
| 5 | Створення бази даних | *26.11.2021* |  |
| 6 | Створення користувачів бази даних | *05.12.2021* |  |
| 7 | Імпорт даних з використанням засобів СУБД | *10.12.2021* |  |
| 8 | Створення запитів до розробленої БД | *17.12.2021* |  |
| 9 | Оптимізація роботи запитів | *20.12.2021* |  |
|  |  |  |  |

**Календарний план**

| **Студент** |  |  | *Полішко А. О.* |
| --- | --- | --- | --- |

(підпис) (прізвище та ініціали)

| **Керівник роботи** |  |  | *Марченко О. І.* |
| --- | --- | --- | --- |

(підпис) (прізвище та ініціали)

**Вступ**

У сьогоденні людям доводиться оперувати величезними об’ємами інформації, яка без належної структури та подання, буде не доступна для використання. Інформація має бути не тільки доступна для використання, але швидко аналізуватися, підлягати зміні та зручній імплементації з технологіями у яких ця інформація використовується. Для таких цілей використовують бази даних (надалі - БД). БД широко використовуються як у колосальних комерційних проектах так і в невеликих рішеннях, як та, що буде моделюватись протягом цієї курсової роботи. При виконанні поставлених задач використовувались такі інструменти :

- MySQL – швидка та сучасна система управління базами даних (надалі - СУБД), використання якої широко поширене;

- draw.io – інструмент для створення діаграм, що надає необхідних функціонал безкоштовно.

# 1 Аналіз предметної області

Потрібно розробити базу даних для підтримки діяльності рекламного агентства.

Основною сутністю є рекламне агентство, яке має наступні атрибути:

- Унікальний номер

- Тип(творче або медійне)

- Мінімальна ціна проекту для співпраці

- Середня ціна однієї години роботи

- Кількість працівників

- Дата заснування

- Розташування головного офісу

- ФІО виконувача обов’язків голови агентства

- Основне середовище публікації реклами(ТБ, інтернет, друковані матеріали і т.д.)

Замовник — юридична чи фізична особа, що замовляє послуги агентства. Його атрибути:

- Унікальний номер

- ПІБ замовника

- Тип замовника ( юридична чи фізична особа )

- Максимальна сума, що виділяється на замовлення

Замовлена послуга — продукт або інша послуга, яку замовляє замовник. Її атрибути:

- Унікальний номер

- Ціна

- Дата початку виконання послуги

- Прогнозований дата закінчення виконання послуги

Офіс, що виконує замовлення — той офіс, в який делегується виконання послуги. Його атрибути:

- Унікальний номер

- Розташування

- Кількість працівників

- Середня ціна години роботи

Керівник виконання замовленого проекту — особа,що є відповідальною за виконання проекту та делегує процес. Його атрибути:

- Унікальний номер

- ПІБ

Посада керівника проект. Її атрибути:

- Унікальний номер

- Назва

- Заробітна плата

Середовище публікації реклами — середовище, в якому планується реалізувати рекламу:

- Унікальний номер

- Прогнозована охоплена аудиторія

- Тип середовища( ТБ, інтернет, друковані матеріали т.д.)

Мета замовленої послуги — описує цілі, для яких послуга рекламується. Її атрибути:

- Унікальний номер

- Тип (благодійність, комерційні наміри тощо)

Додаткові залучені компанії — у процесі реклами можуть залучатись допоміжні компанії. Їх атрибути:

- Унікальний номер

- Назва

- Тип

- Мета, з якою компанія була залучена

Аудиторія реклами — у кожного продукту, чи то товар чи будь-що інше, є цільова аудиторія, яку бажають зацікавити. Її атрибути :

- Унікальний номер

- Прогнозована кількість

- Середній прогнозований вік аудиторії.

Варто передбачити наступні обмеження:

1. Максимальна сума виділена на замовлення замовником має бути не менше 1000 і не більше 1000000

2. Дата початку виконання послуги не може бути більш за прогнозовану дату кінця виконання послуги

3.Кількість працівників у офісі, що виконує замовлення не може бути менше 40 і більше 400

4.Iм’я керівника проекту не довше 100 символів.

5.Кількість аудиторії не може бути менше 100 і більше 100000

6.Вік аудиторії не може бути менше 1 року або більше 150 років.

7.Зарплата керівника проекту не може бути менше 1000 і більше 10000

Підсумок розділу: ми проаналізували детально проаналізували предметну область, та сутності, визначили, що вони будуть виконувати та привели можливі обмеження для підтримки нашої БД

**2 ПОСТАНОВКА ЗАДАЧІ**

Необхідно спроектувати БД для роботи рекламного агентства. Для цього потрібно зробити наступне:

1.проаналізувати предметне середовище;

2.побудувати ER-діаграму та схему БД;

3.створити та заповнити БД записами;

4.розробити скрипти та запити на мові SQL, оптимізувати їх роботу.

В результаті ми створюємо базу даних для підтримки роботи рекламного агентства

**3 Концептуальна модель бази даних**

3.1 Побудувати ER діаграму

Наша БД матиме 10 сутностей. Їх назви та опис зазначені у таблиці 3.1

Таблиця 3.1

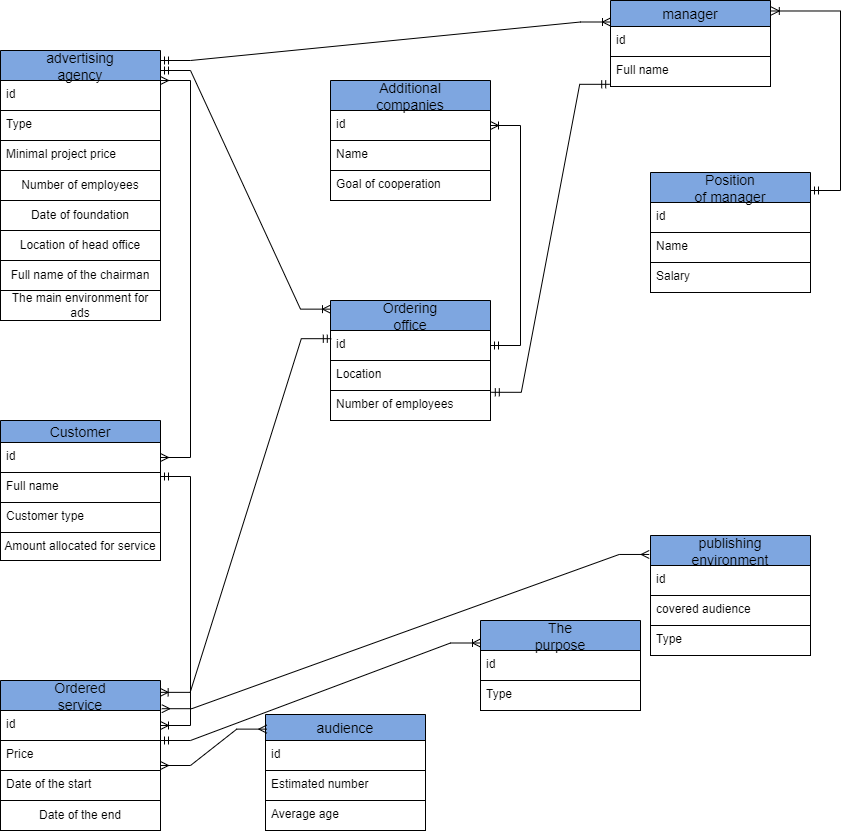
| **Назва** | **Опис** |
| --- | --- |
| рекламне агентство(advertising agency) | Інформує про тип агентства; мінімальну ціну проекту для співпраці;середню ціну однієї години роботи;кількість працівників;дату заснування;розташування головного офісу;ФІО голови агентства; основне середовище публікацій реклами |
| Замовник(Customer) | Інформує про ПІБ замовника;Тип замовника;Максимальна сума, що виділяється на замовлення |
| Замовлена послуга(Ordered service) | Інформує про Ціну;Дата початку виконання послуги;Прогнозований дата закінчення виконання послуги |
| Офіс, що виконує замовлення(Ordering office) | Інформує про Розташування;Кількість працівників;Середня ціна години роботи |
| Керівник виконання замовленого проекту(manager) | Інформує про ПІБ керівника проекту |
| Посада керівника проекту(Position of manager) | Інформує про назву посади;заробітну плату |
| Середовище публікації реклами(publishing environment) | Інформує про Прогнозовану охоплену аудиторію;Тип середовища( ТБ, інтернет, друковані матеріали т.д.) |
| Мета замовленої послуги (The purpose) | Інформує про Тип (благодійність, комерційні наміри тощо) |
| Додаткові залучені компанії(Additional companies) | Інформує про Назву;Тип;Мету, з якою компанія була залучена |
| Аудиторія реклами(audience) | Інформує про Прогнозовану кількість;Середній прогнозований вік аудиторії. |

Зв’язки між сутностями наведені у таблиці 3.2

Таблиця 3.2

| рекламне агентство | Замовник | Багато до багатьох |
| --- | --- | --- |
| рекламне агентство | Офіс, що виконує замовлення | Один до багатьох |
| рекламне агентство | Керівник виконання замовленого проекту | Один до багатьох |
| Замовник | Замовлена послуга | Один до багатьох |
| Замовлена послуга | Аудиторія реклами | Багато до багатьох |
| Замовлена послуга | Мета замовленої послуги | Один до багатьох |
| Замовлена послуга | Середовище публікації реклами | Багато до багатьох |
| Замовлена послуга | Офіс, що виконує замовлення | Багато до одного |
| Офіс, що виконує замовлення | Керівник виконання замовленного проекту | Один до одного |
| Офіс, що виконує замовлення | Додаткові залучені компанії | Один до багатьох |
| Керівник виконання замовленного проекту | Посада керівника проекту | Багато до одного |

З отриманих таблиць можемо побудувати ER діаграму. Вона зображена на рисунку 3.1

Рис 3.1

Висновок розділу: ми описали таблиці та навели концептуальну ER модель

**4 Логічна Модель Бази Даних**

Для втілення зв’язків багато до багатьох потрібно додати декілька допоміжних таблиць:

- company\_customer – для зв’язку агентства та замовника

- service\_environment – для зв’язку послуги і середовища публікації

- service\_audience – для зв’язку послуги і аудиторії

Схеми таблиць зображені на таблицях 4.1.1 -

Таблиця 4.1.1 — Схема агентства

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Type | var\_char |
| Minimal\_project\_price | int |
| Number of employees | int |
| Location of head office | var\_char |
| Full\_name\_of\_the\_chairman | var\_char |
| The\_main\_environment\_for\_ads | var\_char |
| id\_customer(FK) | int |

Таблиця 4.1.2 — Схема замовника

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Full\_name | var\_char |
| Customer\_type | var\_char |
| Amount\_allocated\_for\_service | int |
| id\_agency(FK) | int |

Таблиця 4.1.3 Схема замовленої послуги

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Price\_of\_service | int |
| Date\_of\_the\_start | date |
| Date\_of\_the\_end | date |
| id\_customer(FK) | int |
| id\_audience(FK) | int |
| id\_office(FK) | int |
| id\_environment(FK) | int |

Таблиця 4.1.4 Схема аудиторії

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Estimated\_number | int |
| Average\_age | int |
| id\_service(FK) | int |

Таблиця 4.1.5 Схема мети

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Type | var\_char |
| id\_service(FK) | int |

Таблиця 4.1.6 Схема середовища публікації

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Type | var\_char |
| Covered\_audience | var\_char |
| id\_service(FK) | int |

Таблиця 4.1.7 Схема офісу, що виконує замовлення

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Number\_of\_employees | int |
| Location | var\_char |
| id\_manager(FK) | int |

Таблиця 4.1.8 Схема додаткових компаній

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Goal\_of\_cooperation | var\_char |
| Name | var\_char |
| id\_orderingOffice(FK) | int |

Таблиця 4.1.9 Схема керівника виконання проекту

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| Full\_name | var\_char |
| id\_orderingOffice(FK) | int |
| id\_position (FK) | int |
| id\_agency(FK) | int |

Таблиця 4.1.10 Схема посади керівника проекту

| **Атрибут** | **Тип даних** |
| --- | --- |
| id(PK) | int |
| name | var\_char |
| salary | double |

Допоміжна таблиця 4.1.11 Схема зв’язку замовника і агентства

| **Атрибут** | **Тип даних** |
| --- | --- |
| id\_agency(PK)(FK) | int |
| id\_customer(PK)(FK) | int |

Допоміжна таблиця 4.1.12 Схема зв’язку замовлення та аудиторії

| **Атрибут** | **Тип даних** |
| --- | --- |
| id\_service(PK)(FK) | int |
| id\_audience(PK)(FK) | int |

Допоміжна таблиця 4.1.12 Схема зв’язку замовлення та середовища

| **Атрибут** | **Тип даних** |
| --- | --- |
| id\_service(PK)(FK) | int |
| id\_environment(PK)(FK) | int |

Тоді кінцева ER діаграма виглядатиме так(рис 4.1)

рис 4.1

Висновок розділу: в результаті цього розділу ми детально описали поля таблиць та навели кінцевий вигляд логічної моделі БД з таблицями для підтримки звязків багато до батьох

**5 Реалізація Бази Даних**

5.1 Проектування БД

База даних була спроектована за допомогою MySQL Workbench через зручний інтерфейс.

Нижче наведений код для створення таблиць та їх полів

**create table advertesingAgency (**

**id int not null auto\_increment,**

**type\_ varchar(64),**

**minimalProjectPrice int,**

**numberOfEmployees int not null,**

**dateOfFoundation date,**

**locationOfHeadOffice varchar(64),**

**fullNameOfTheChairman varchar(64),**

**theMainEnvironmentForAds varchar(64),**

**primary key(id),**

**id\_customer int not null**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table customer (**

**id int not null auto\_increment,**

**fullName varchar(64),**

**customerType varchar(64),**

**amountAllocatedForService int,**

**primary key(id),**

**id\_agency int not null**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table agency\_customer (**

**id\_agency int not null,**

**id\_customer int not null,**

**primary key(id\_agency,id\_customer),**

**constraint id\_agency foreign key(id\_agency) references advertesingAgency(id) on update cascade on delete cascade,**

**constraint id\_customer foreign key(id\_customer) references customer(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table orderingOffice(**

**id int not null auto\_increment,**

**location varchar(64),**

**numberOfEmployees int,**

**primary key(id)**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table orderedService (**

**id int not null auto\_increment,**

**price int not null,**

**dateOfTheStart date,**

**dateOfTheEnd date,**

**primary key(id),**

**id\_customer int not null,**

**constraint fk2\_id\_customer foreign key(id\_customer) references customer(id) on update cascade on delete cascade,**

**id\_office int not null,**

**constraint fk1\_id\_office foreign key(id\_office) references orderingoffice(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table audience (**

**id int not null auto\_increment,**

**estimatedNumber int,**

**avrageAge int,**

**primary key(id),**

**id\_service int not null**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table audience\_service (**

**id\_service int not null,**

**id\_audience int not null,**

**primary key(id\_service,id\_audience),**

**constraint fk1\_id\_service foreign key(id\_service) references orderedService(id) on update cascade on delete cascade,**

**constraint fk1\_id\_audience foreign key(id\_audience) references audience(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table thePurpose (**

**id int not null auto\_increment,**

**type\_ varchar(64),**

**primary key(id),**

**id\_service int not null,**

**constraint fk2\_id\_service foreign key(id\_service) references orderedService(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table publishingEnvironment (**

**id int not null auto\_increment,**

**coveredAudience int,**

**type\_ varchar(64),**

**primary key(id)**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table service\_environment (**

**id\_service int not null,**

**id\_environment int not null,**

**primary key(id\_service,id\_environment),**

**constraint fk3\_id\_service foreign key(id\_service) references orderedService(id) on update cascade on delete cascade,**

**constraint fk1\_id\_environment foreign key(id\_environment) references publishingenvironment(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table additionalCompanie (**

**id int not null auto\_increment,**

**name\_ varchar(64),**

**goalOfCooperation varchar(64),**

**primary key(id),**

**id\_office int not null,**

**constraint fk2\_id\_office foreign key(id\_office) references orderingoffice(id) on update cascade on delete cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table positionOfManager (**

**id int not null auto\_increment,**

**name\_ varchar(64),**

**salary int,**

**primary key (id)**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**create table manager (**

**id int not null auto\_increment,**

**fullName varchar(64),**

**primary key (id),**

**constraint fk1\_id foreign key(id) references orderingoffice(id) on update cascade on delete cascade,**

**id\_agency int not null,**

**constraint fk\_id\_agency foreign key(id\_agency) references advertesingagency(id) on delete cascade on update cascade,**

**id\_position int not null,**

**constraint fk\_id\_position foreign key(id\_position) references positionOfManager(id) on delete cascade on update cascade**

**)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

На рисунку 5.1 зображена схема побудована в результаті виконання описаного вище коду.

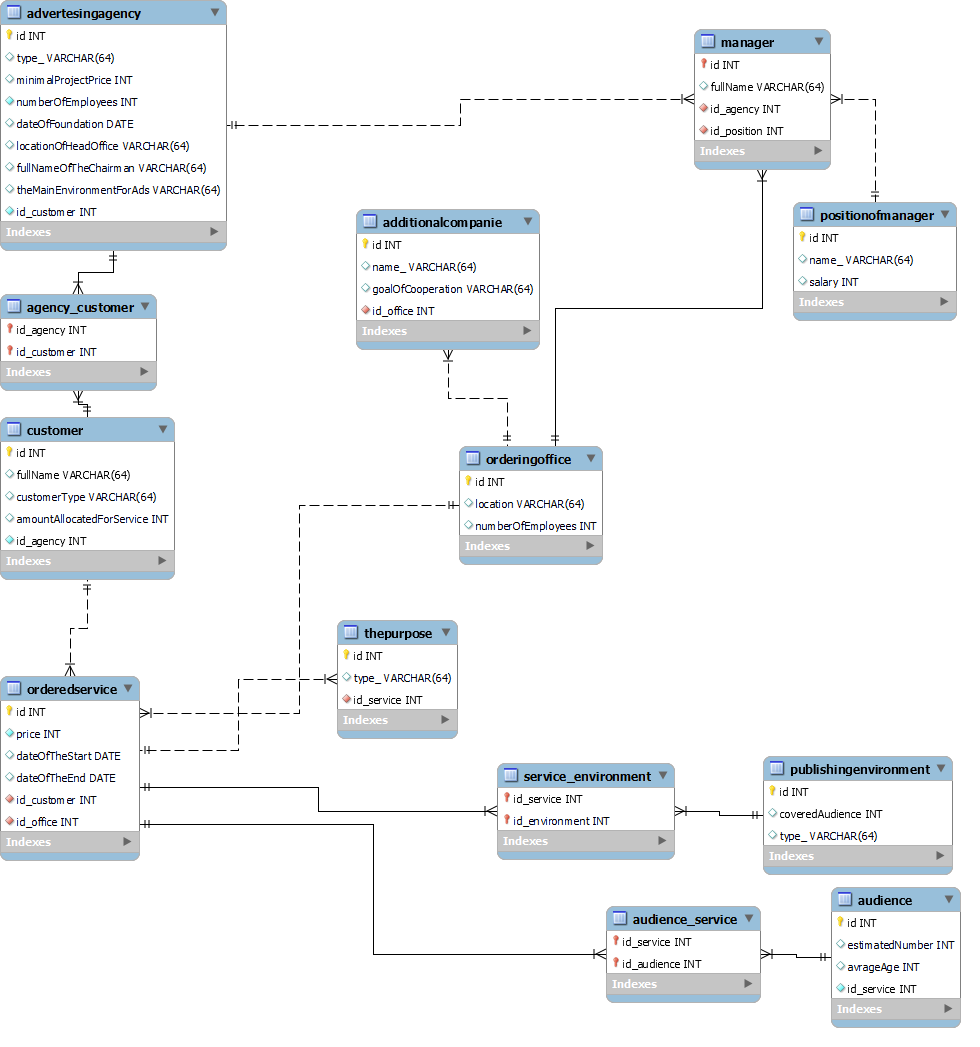


рис 5.1

5.2 Створення обмежень

1)Обмеження на довжину імені керівника проекту

**alter table manager**

**add constraint checkLongevityOfTheName check (CHAR\_LENGTH(fullName)<100);**

2)Обмеження на мінімальну ціну проекту

**alter table advertesingagency**

**add constraint checkMinPrice check(minimalProjectPrice>=1000 and minimalProjectPrice<=1000000);**

3)Обмеження на зарплату керівників проектів

**alter table positionOfManager**

**add constraint checkSallary check(salary>=1000 and salary<=10000);**

4)Обмеження на розмір удиторії

**alter table audience**

**add constraint checkNumber check(estimatedNumber>=100 and estimatedNumber<=100000);**

5)Обмеження на кільксть працівників

**alter table orderingoffice**

**add constraint checkNumOfWorkers check(numberOfEmployees>=40 and numberOfEmployees<=400);**

6)Обмеження на вік аудиторії

**alter table audience**

**add constraint checkAge check(averageAge>=1 and averageAge<=150);**

7)Обмеження на дати проектів

**alter table orderedService**

**add constraint checkDate check(dateOfTheEnd>dateOfTheStart);**

5.3 Створення користувачів

Для підвищення зручності та безпеки нам потрібно створити користувачів, у кожного з яких буде свій діапазон прав та певні обмеження користування системою. Не кожен користувач має мати права для повної взаємодії з БД, тому ми маємо розподілити ці права по користувачам.

Ми можемо це зробити за допомогою SQL скриптів.

Створимо користувача **kursova\_admin1** у якого будуть права на будь які зміни у базі даних **kursova**:

**create user kursova\_admin1@localhost identified by "admin123";**

Користувача **kursova\_customer1** у якого будуть права лише на перегляд даних у базі даних **kursova:**

**create user kursova\_customer1@localhost identified by "customer123";**

Нам потрібно створити відповіді ролі, яким будуть надані бажані нам права та надати ці ролі користувачам:

**create role admin, customer;**

**grant select on kursova.\* to customer;**

**grant all on kursova.\* to admin;**

**grant customer to kursova\_customer1@localhost;**

**grant admin to kursova\_admin1@localhost;**

Таблиця 5.3.1 з логінами, паролями та правами користувачів

| Логін | Пароль | Права |
| --- | --- | --- |
| **kursova\_admin1** | **admin123** | усі |
| **kursova\_customer1** | **customer1**23 | права на перегляд на перегляд |

5.4 Імпорт записів до БД

Обрана СУБД - MySQL підтримує імпорт даних. Завчасно згенеровані файли розширення .csv можна імпортувати за допомогою MySQL Workbench, якщо слідувати наступним крокам.

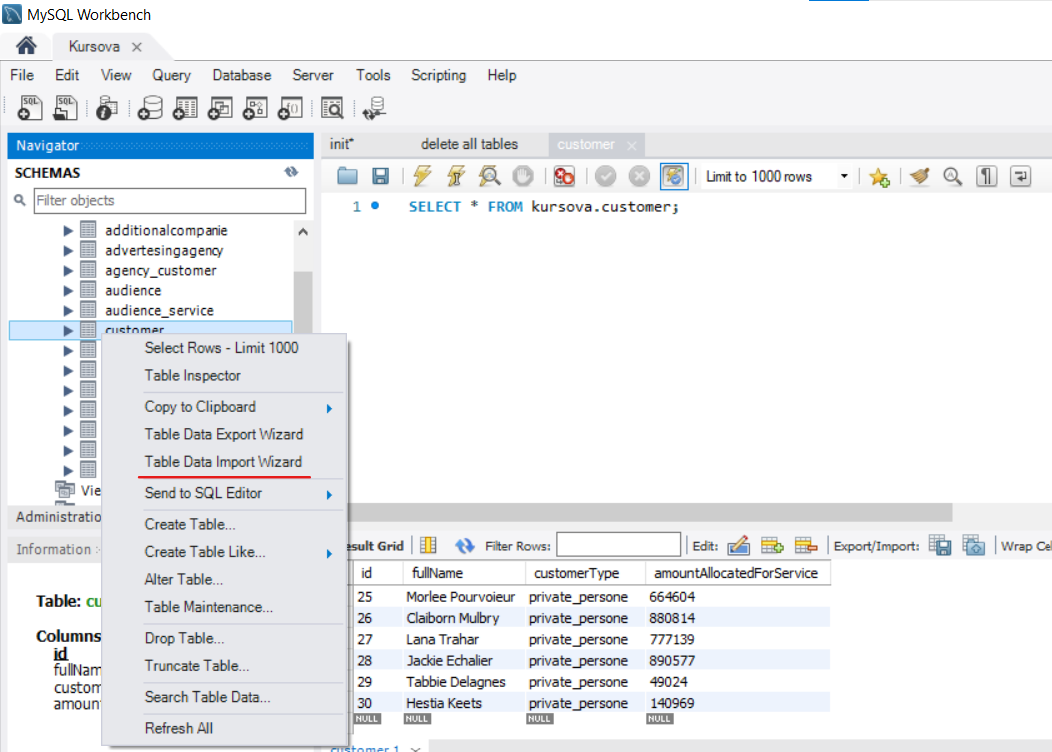


рис. 5.4.1

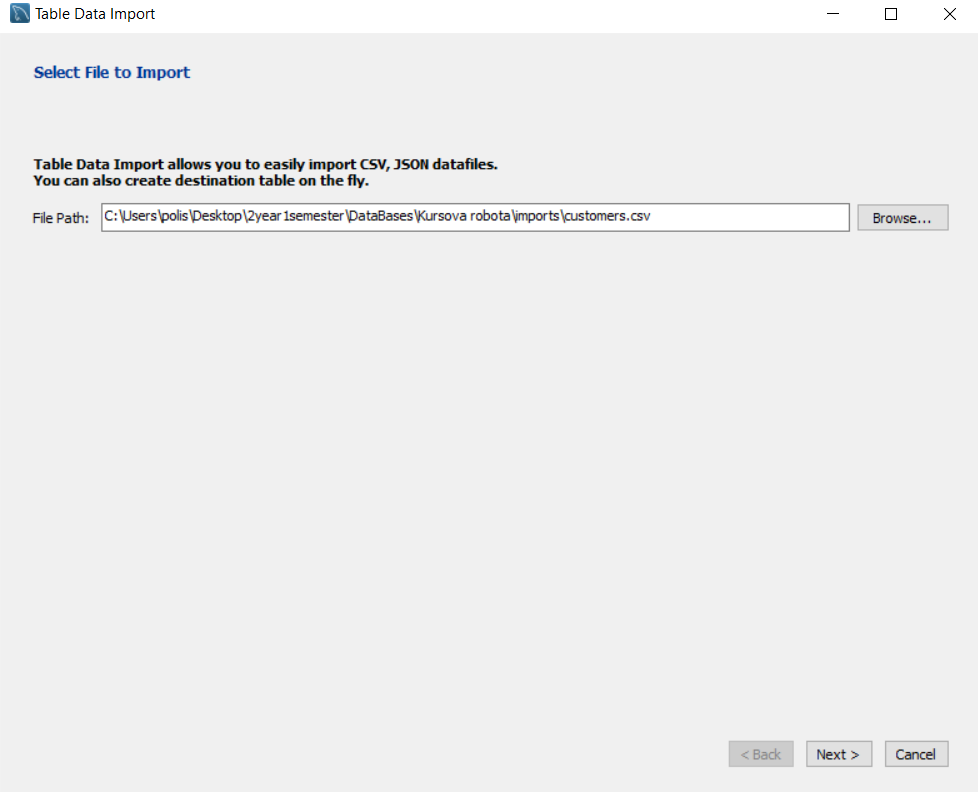


рис 5.4.2

Далі необхідно налаштувати параметри імпорту і запустити його.

5.5 Реалізація запитів

1. Обрати менеджера з конкретним ім’ям

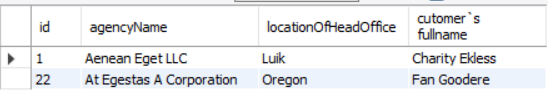
**select advertesingagency.id,advertesingagency.agencyName,advertesingagency.locationOfHeadOffice, customer.fullname as "cutomer`s fullname"**

**from advertesingagency**

**join agency\_customer on advertesingagency.id = agency\_customer.id\_agency**

**join customer on customer.id = agency\_customer.id\_customer**

**where advertesingagency.agencyName like("A%")**

****

2)Обрати аудиторію у межах середнього віку

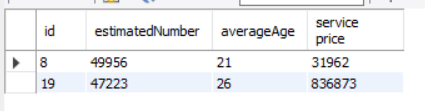
**select audience.\*,orderedservice.price as "service price"**

**from audience**

**join audience\_service on audience\_service.id\_audience = audience.id**

**join orderedservice on orderedservice.id = audience\_service.id\_service**

**where audience.averageAge between 18 and 28**



3)Обрати замовників з сумою, яка може покрити витрати

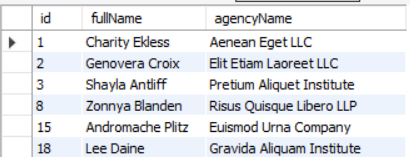
**select customer.id,customer.fullName,advertesingagency.agencyName**

**from customer**

**join agency\_customer on customer.id = agency\_customer.id\_customer**

**join advertesingagency on advertesingagency.id = agency\_customer.id\_agency**

**where customer.amountAllocatedForService > advertesingagency.minimalProjectPrice**

****

4)Обрати замовників з конкретним id

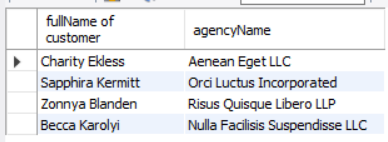
**select customer.fullName as "fullName of customer", advertesingagency.agencyName**

**from customer**

**join agency\_customer on customer.id = agency\_customer.id\_customer**

**join advertesingagency on advertesingagency.id = agency\_customer.id\_agency**

**where customer.id in (1,5,8,20)**



5)Відобразити скільки разів користувач може оплатити послугу,якщо не 0

**select customer.id,floor(customer.amountAllocatedForService/orderedservice.price) as "Paid services"**

**from customer**

**join orderedservice on orderedservice.id\_customer = customer.id**

**where floor(customer.amountAllocatedForService/orderedservice.price) > 0**



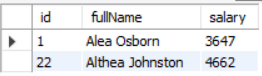
6)Обрати менеджерів з специфічним ім’ям за допомогою regexp

**select manager.id,manager.fullName,positionofmanager.salary**

**from manager**

**join positionofmanager on positionofmanager.id = manager.id\_position**

**where manager.fullName regexp '^A' and manager.fullName regexp 'n$'**



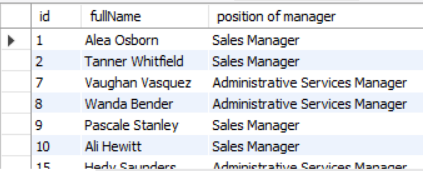
7)Обрати менеджера з посадою, що входить в перелічений список

**select manager.id,manager.fullName, positionofmanager.name\_ as "position of manager"**

**from manager**

**join positionofmanager on manager.id\_position = positionofmanager.id**

**where positionofmanager.name\_ in ('Sales Manager','Administrative Services Manager')**



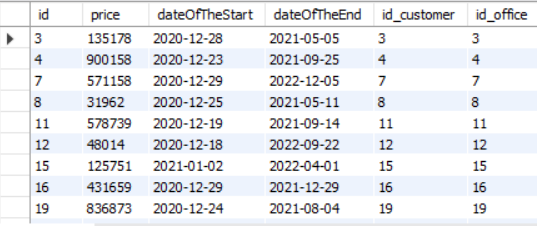
8)Обрати послуги з некомерційною метою

**select orderedservice.\***

**from orderedservice**

**join thepurpose on thepurpose.id\_service=orderedservice.id**

**where thepurpose.type\_ != "commercial"**



9)Обрати офіси з певним розташуванням

**select orderingoffice.id, orderingoffice.location, orderingoffice.numberOfEmployees,advertesingagency.agencyName**

**from orderingoffice**

**join advertesingagency on advertesingagency.id = orderingoffice.id\_agency**

**where orderingoffice.location = "Boston"**



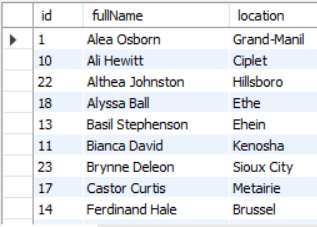
10)Обрати розташування офісів менеджерів і відсортувати за ім’ям

**select manager.id,manager.fullName,orderingoffice.location**

**from manager**

**join orderingoffice on orderingoffice.id = manager.id**

**order by fullName**



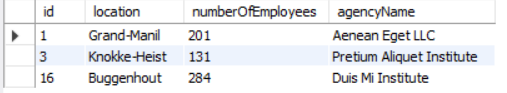
11)Обрати офіси з специфічною кількістю працівників

**select orderingoffice.id, orderingoffice.location, orderingoffice.numberOfEmployees,advertesingagency.agencyName**

**from orderingoffice**

**join advertesingagency on advertesingagency.id = orderingoffice.id\_agency**

**where orderingoffice.numberOfEmployees in (201,131,284);**



12)Обрати офіси з специфічними підсобними компаніями

**select orderingoffice.\*,additionalcompanie.name\_ as "additionalcompanie"**

**from additionalcompanie**

**join orderingoffice on additionalcompanie.id\_office = orderingoffice.id**

**where additionalcompanie.goalOfCooperation = "transportation" or additionalcompanie.goalOfCooperation = "security\_services"**



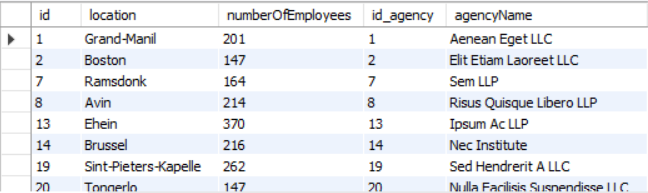
13) Обрати офіси, які підпорядковуються агентствам, якого типу

**select orderingoffice.\*,advertesingagency.agencyName**

**from orderingoffice**

**join advertesingagency on advertesingagency.id = orderingoffice.id\_agency**

**where advertesingagency.type\_ = "Full-service"**

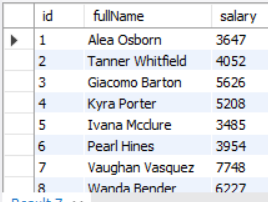
****

14)Отримати імена менеджерів і їх зарплати

**select manager.id,manager.fullName,positionofmanager.salary**

**from manager**

**join positionofmanager on manager.id\_position = positionofmanager.id**



15)Отримати тільки ті послуги у яких аудиторія буде більше зазначеної

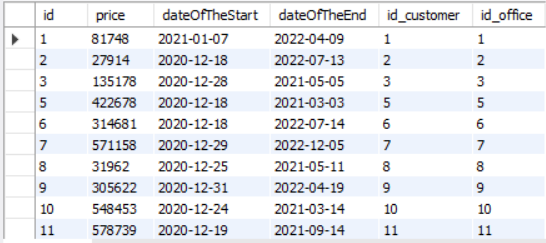
**select orderedservice.\***

**from orderedservice**

**join service\_environment on orderedservice.id = service\_environment.id\_service**

**join publishingenvironment on publishingenvironment.id = service\_environment.id\_environment**

**where coveredAudience>2000**



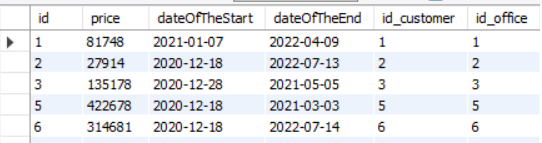
16) Отримати замовлення, де офіси , які над ними працюють, налічують більше зазначеної кількості працівників

**select orderedservice.id,orderedservice.price,dateOfTheStart,dateOfTheEnd**

**from orderedservice**

**join orderingoffice on orderingoffice.id = orderedservice.id\_office**

**where orderingoffice.numberOfEmployees > 100**



17)Визначити некомерційні замовлення, що публікуватимутсья в інтернеті

**select orderedservice.\***

**from orderedservice**

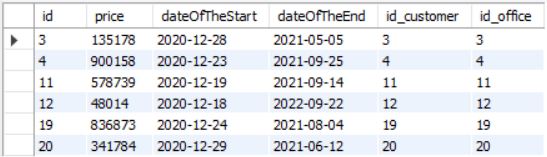
**join thepurpose on thepurpose.id\_service=orderedservice.id**

**join service\_environment on service\_environment.id\_service = orderedservice.id**

**join publishingenvironment on publishingenvironment.id = service\_environment.id\_environment**

**where thepurpose.type\_ = "noncommercial"**

**and publishingenvironment.type\_ in ('Internet')**



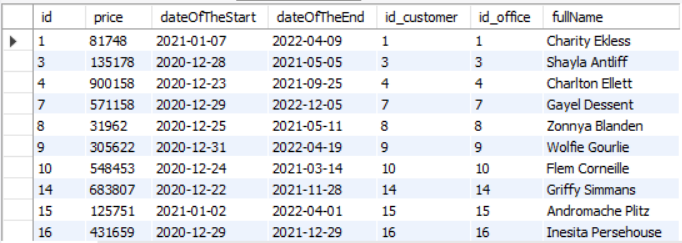
18)Визначити сервіси виконання яких починається з певної дати

**select orderedservice.\*,customer.fullName**

**from orderedservice**

**join customer on customer.id = orderedservice.id\_customer**

**where orderedservice.dateOfTheStart >"2020-12-19"**



19)Визначити який замовник відповідає якому агентству

**select customer.\*,advertesingagency.\***

**from customer**

**join agency\_customer on customer.id = agency\_customer.id\_customer**

**join advertesingagency on advertesingagency.id = agency\_customer.id\_agency**

****

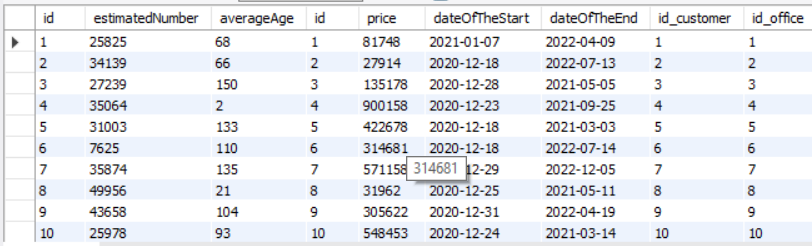
20) Визначити відповідну до замовлення аудиторію

**select audience.\*,orderedservice.\***

**from audience**

**join audience\_service on audience.id = audience\_service.id\_audience**

**join orderedservice on orderedservice.id = audience\_service.id\_service**



5.7 Реалізація тригерів

1)Тригер при оновленні, щоб відсікати небажані призначення компаній

**delimiter //**

**create trigger updateAditionalComp**

**before update**

**on additionalcompanie**

**for each row**

**begin**

**if new.goalOfCooperation not in ("security\_services",'medical\_services','transportation')**

**then set new.goalOfCooperation = "different type";**

**end if;**

**end;//**

**delimiter ;**

2)Тригер при вставці, щоб виправляти зарплату

**delimiter //**

**create trigger checkPositionSallary before insert on positionofmanager**

**for each row if new.salary<=0**

**then set new.salary=1000;**

**end if;//**

**delimiter ;**

3)Тригер для підрахунку кількості вствлених користувачів

**set @num = 0;**

**#drop trigger custNum;**

**create trigger custNum after insert on customer**

**for each row set @num = @num + 1;**

**select @num as "number of added customers" ;**

5.8 Реалізація індексів

1)Індекс для назв агентств

**create index agencyName\_indx**

**on advertesingagency(agencyName);**



рис 5.8.1 До оптимізації



рис 5.8.2 Після оптимізації

2)Індекс на імя замовника

**create index fullname\_indx**

**on customer(fullName);**

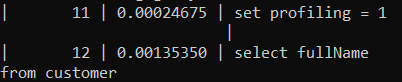


рис 5.8.1 До оптимізації

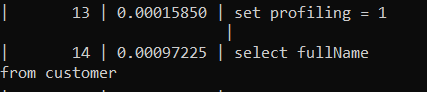


рис 5.8.2 Після оптимізації

5.9 Реалізація функцій

1)Перевірка чи бюджет більше за мінімальну ціну проекту

**create function checkPrices(minimalP int,budget int)**

**returns bool**

**return minimalP<=budget;**

2)Перевірка назв агентств

**delimiter //**

**create function agencyNameCheck(agencyName varchar(64))**

**returns varchar(64)**

**begin**

**declare result varchar(64);**

**if agencyName like('A%') then set result = agencyName;**

**else set result = "wrong";**

**end if;**

**return (result);**

**end//**

**delimiter ;**

3)Перевірка чи входить індекс в сукупність

**create function checkId (id int)**

**returns bool**

**return id in (1,5,8,20)**

5.10 Реалізація представлень

Для кожного запиту було реалізоване представлення

1)

**create view agencyName**

**as**

**select advertesingagency.id,advertesingagency.agencyName,advertesingagency.locationOfHeadOffice, customer.fullname as "cutomer`s fullname"**

**from advertesingagency**

**join agency\_customer on advertesingagency.id = agency\_customer.id\_agency**

**join customer on customer.id = agency\_customer.id\_customer**

**where advertesingagency.agencyName like("A%")**

**2)**

**create view audieceAgeRange**

**as**

**select audience.\*,orderedservice.price as "service price"**

**from audience**

**join audience\_service on audience\_service.id\_audience = audience.id**

**join orderedservice on orderedservice.id = audience\_service.id\_service**

**where audience.averageAge between 18 and 28**

**3)**

**create view cutomers\_That\_Can\_Pay**

**as**

**select customer.id,customer.fullName,advertesingagency.agencyName**

**from customer**

**join agency\_customer on customer.id = agency\_customer.id\_customer**

**join advertesingagency on advertesingagency.id = agency\_customer.id\_agency**

**where customer.amountAllocatedForService > advertesingagency.minimalProjectPrice**

**4)**

**create view specific\_id\_cutomer**

**as**

**select customer.fullName as "fullName of customer", advertesingagency.agencyName**

**from customer**

**join agency\_customer on customer.id = agency\_customer.id\_customer**

**join advertesingagency on advertesingagency.id = agency\_customer.id\_agency**

**where customer.id in (1,5,8,20)**

**5)**

**create view number\_paid\_services\_per\_customer**

**as**

**select customer.id,floor(customer.amountAllocatedForService/orderedservice.price) as "Paid services"**

**from customer**

**join orderedservice on orderedservice.id\_customer = customer.id**

**where floor(customer.amountAllocatedForService/orderedservice.price) > 0**

**6)**

**create view specific\_managers\_name**

**as**

**select manager.id,manager.fullName,positionofmanager.salary**

**from manager**

**join positionofmanager on positionofmanager.id = manager.id\_position**

**where manager.fullName regexp '^A' and manager.fullName regexp 'n$'**

**7)**

**create view select\_managers\_with\_certain\_position**

**as**

**select manager.id,manager.fullName, positionofmanager.name\_ as "position of manager"**

**from manager**

**join positionofmanager on manager.id\_position = positionofmanager.id**

**where positionofmanager.name\_ in ('Sales Manager','Administrative Services Manager')**

**8)**

**create view select\_non\_comercial\_servicies**

**as**

**select orderedservice.\***

**from orderedservice**

**join thepurpose on thepurpose.id\_service=orderedservice.id**

**where thepurpose.type\_ != "commercial"**

**9)**

**create view select\_office\_of\_manager**

**as**

**select manager.id,manager.fullName,orderingoffice.location**

**from manager**

**join orderingoffice on orderingoffice.id = manager.id**

**order by fullName**

**10)**

**create view select\_office\_with\_certain\_location**

**as**

**select orderingoffice.id, orderingoffice.location, orderingoffice.numberOfEmployees,advertesingagency.agencyName**

**from orderingoffice**

**join advertesingagency on advertesingagency.id = orderingoffice.id\_agency**

**where orderingoffice.location = "Boston"**

Висновок розділу:

У цьому розділі ми продемонстрували перенесення БД у код, створення обмежень,функцій, запитів, індексів та представлення

**6 Висновки**

При виконанні цієї курсової роботи було спроектовано базу даних рекламного агентства, були реалізовані та проаналізовані такі аспекти мови програмування SQL, як індекси, представлення, робота з функціями, обмеження та dml запити.

Як результат ми створили робочу базу даних для підтримки рекламного агентства

**7 Список використаної літератури**

1.<https://www.mysqltutorial.org/> - опис великого об’єму функціоналу

2.<https://generatedata.com/generator> - сервіс для генерації даних у .csv

3.<https://stackoverflow.com/> - використане для знаходження деяких рішень

4.<https://dev.mysql.com/doc/> - документація використаного СУБД