МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІСЬКА ПОЛІТЕХНІКА»

Кафедра ІСМ



**Звіт**

До лабораторної роботи №3

З дисципліни:

«Організація сховищ та просторів даних»

На тему:

**«Проектування логічної структури сховища даних з архітектурою зведення даних.**»

*Виконав:*

*студент групи КН-47*

Шандра Олег

*Прийняв:*

*доцент* Лозицький О.А.

Львів 2018

**Мета роботи:** Вивчення порядку, методів та засобів проектування і побудови сховища даних з архітектурою зведення даних та оцінка часу виконання запитів.

**Завдання:**

Порядок виконання роботи

0. Модифікація структури сховища даних КІФ до архітектури зведення даних.

Зміст завдання: сховище даних з лабораторної роботи 1 модифікувати до архітектури зведення даних.

1. Модифікація структури сховища даних зірка до архітектури зведення даних.

Зміст завдання: сховище даних з лабораторної роботи 2 модифікувати до архітектури зведення даних.

2. Висновки щодо ефективності архітектур.

Зміст завдання: визначити час виконання запитів до сховища даних, розроблених у першому та другому завданнях. Визначити переваги та недоліки архітектури зведення даних на основі аналізу часу виконання запитів.

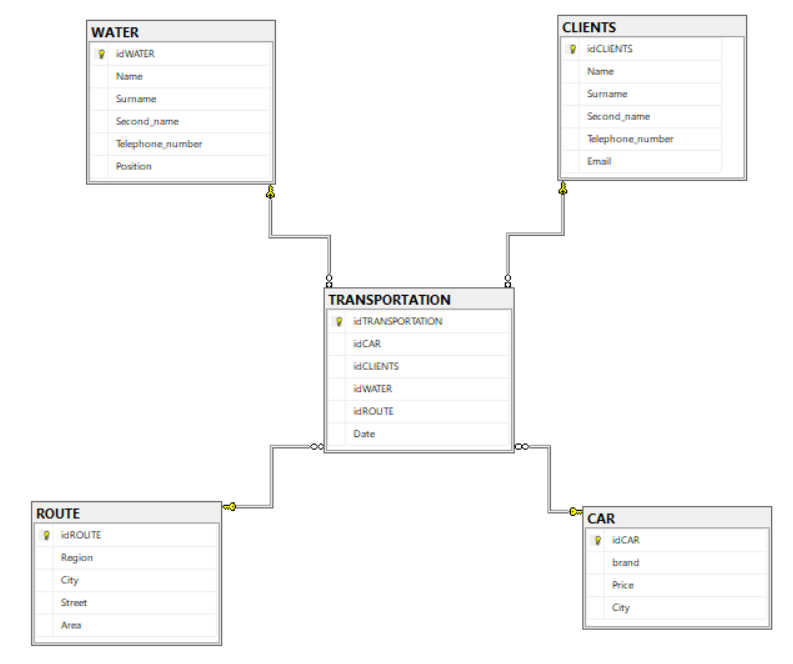
**Теоретичні відомості:**

Зведення даних (Data Vault) (ЗД) – предметно орієнтована, історична і унікально зв'язана множина нормалізованих таблиць, які підтримують одну або більше функціональних предметних областей. Це – гібридний підхід, що поєднує кращі особливості 3-ої нормальної форми (3НФ) і схеми «зірка». У лабораторінй роботі № 1 подано ориґінальну 3НФ модель, пристосовану до архітектури сховищ даних. Одна особливо складна проблема очевидна, коли значення часу-дати розміщена в первинному ключ таблиці батька (див рис. 1). Якщо у таблицю батька додано новий запис, то ця зміна викликає примусове каскадування вниз через всі підлеглі табличні структури. Також, коли новий кортеж вставлений з наявним ключем батька (єдине поле зміни - мітка часу дати), всі кортежі-діти повинні бути переприсвоєні до нового ключа батька. Цей ефект каскадування стає все помітнішим при збільшенні кількості даних. Усе це призводить до неможливості підтримки роботи сховищ даних.

Тому подальшим розвитком була схема «зірка», яка вимагала денормалізації, але водночас була краще пристосована до виконання аналітичних завдань. Модель «зірка» добре працює для швидкого подання багатовимірної інформації для певних груп кінцевих користувачів. Її перевагами є: багатовимірний аналіз, сумарні звіти (графи, пошук мінімуму, максимуму, зміна рівнів аґреґування). Проте, як виявилося, така модель не є гнучкою, оскільки її структура є незмінною, жорстко форматованою. Також вона не здатна забезпечити статистичний аналіз. Схема «зірка» здатна до виконання операції «slice and dice» для окремого користувача або групи користувачів. Також недоліком є неможливість подання усього масиву інформації у вигляді довільних звітів. Нарешті, модель дуже надмірна і важка для здійснення змін у структурі.

Зведення даних передбачає певну нормалізацію даних, що видозмінена для потреб сховищ даних. Ця архітектура використовує такі модельовані методи: зв’язок багато-до-багатьох, цілісність зв’язків, мінімально надмірні набори даних. Ці методи роблять модель зведення даних гнучкою, поширюваною і послідовною. На рис. 2 подано модель зв’язків між сутностями предметної області «Торговельна фірма», яка дозволяє перейти до моделі СД зведення даних.

**Хід виконання завдання:**



*Рис. 1. Схема сховища даних типу зірка.*

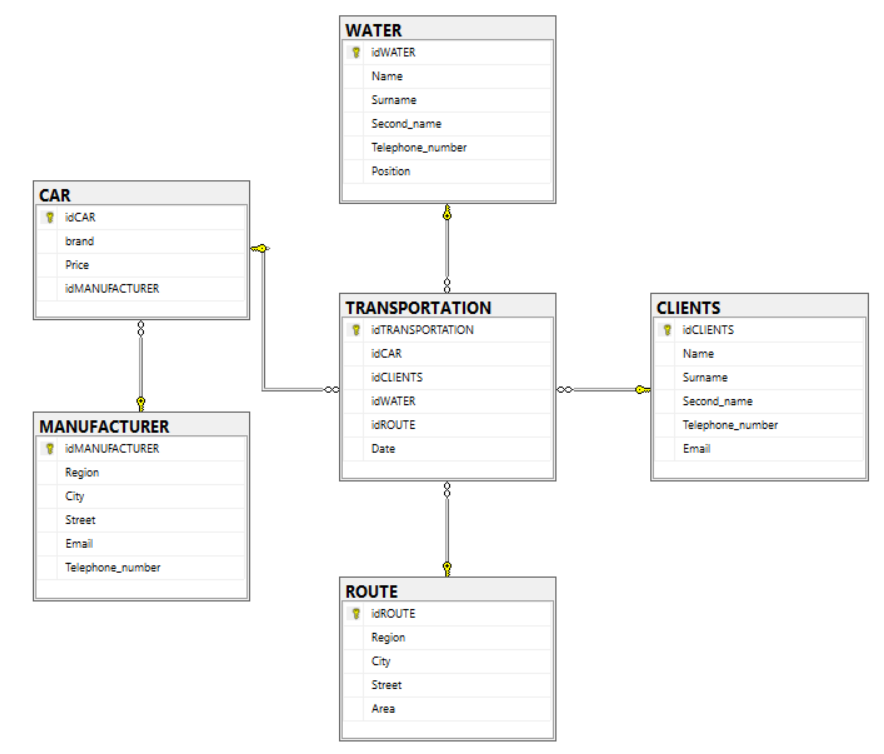
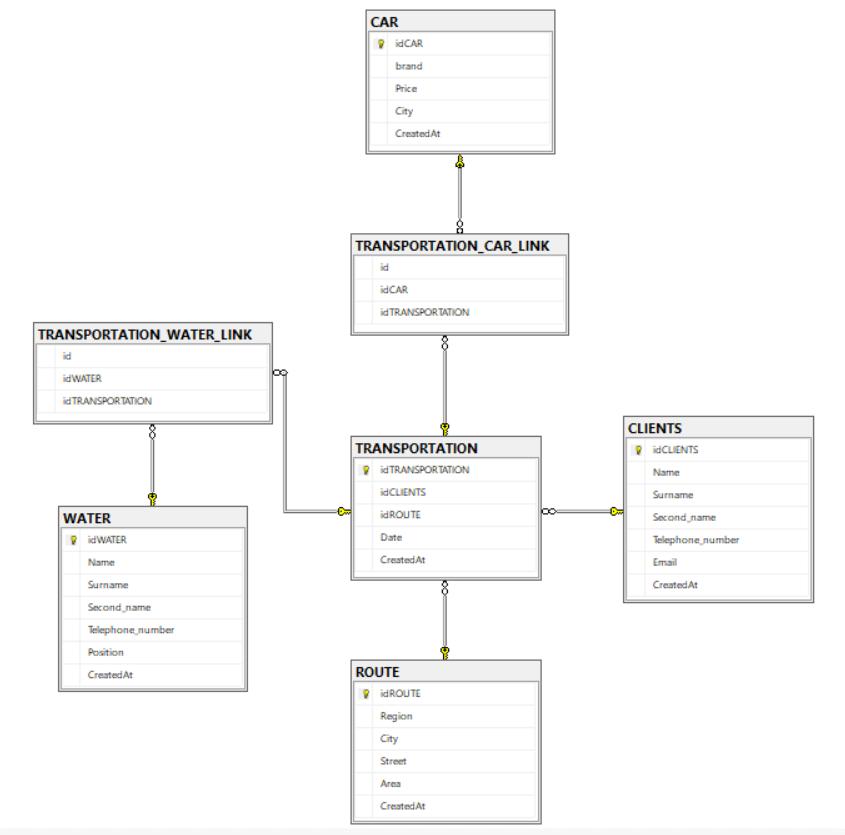
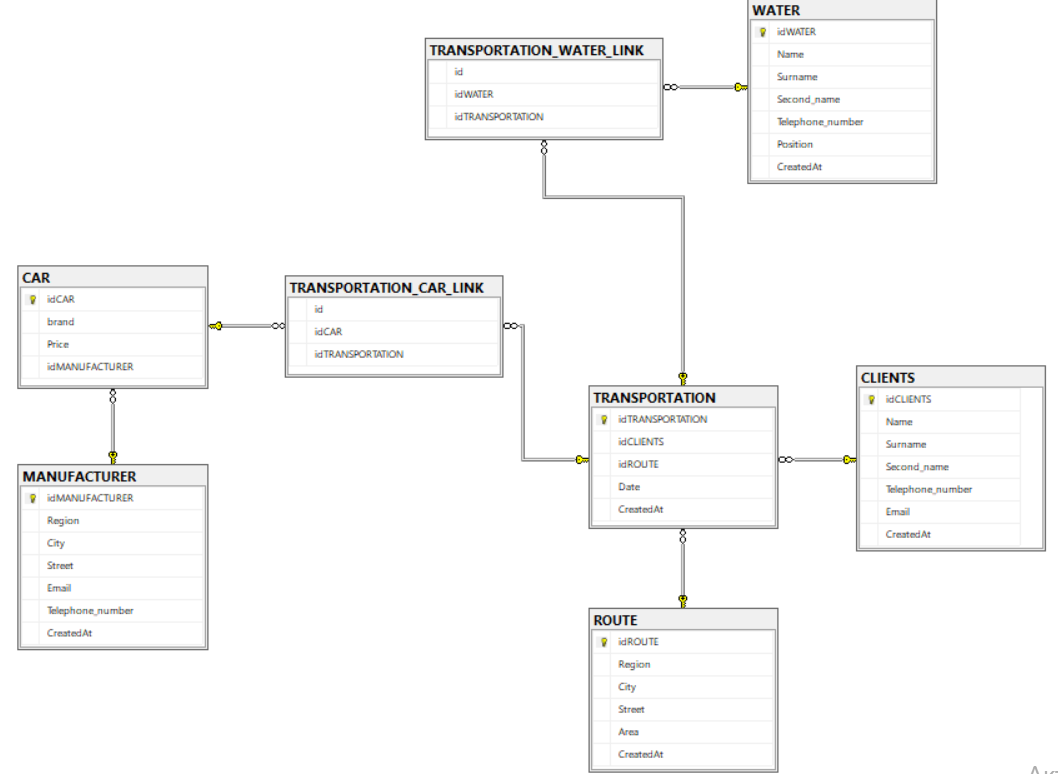


Рис. 2. Схема сховища даних типу сніжинка.



*Рис. 3. Схема сховища даних типу зірка після зведення даних.*



*Рис. 4. Схема сховища даних типу сніжинка після зведення даних*.

IF EXISTS (SELECT \* FROM sys.objects WHERE type = 'P' AND name = 'GETFORMS')

BEGIN

DROP PROC GETFORMS

END

GO

create procedure GETFORMS

as

select \*

from TRANSPORTATION

INNER JOIN CLIENTS on TRANSPORTATION.IDCLIENTS = CLIENTS.IDCLIENTS

INNER JOIN TRANSPORTATION\_WATER\_LINK on TRANSPORTATION.idTRANSPORTATION = TRANSPORTATION\_WATER\_LINK.idTRANSPORTATION

INNER JOIN WATER on TRANSPORTATION\_WATER\_LINK.IDWATER = WATER.idWATER

INNER JOIN TRANSPORTATION\_CAR\_LINK on TRANSPORTATION.idTRANSPORTATION = TRANSPORTATION\_CAR\_LINK.idTRANSPORTATION

INNER JOIN CAR on TRANSPORTATION\_CAR\_LINK.IDCAR = CAR.IDCAR

INNER JOIN ROUTE on TRANSPORTATION.IDROUTE = ROUTE.IDROUTE

*Рис. 5. SQL-скрипт, що отримує дані зі сховища даних типу зірка після зведення даних.*

IF EXISTS (SELECT \* FROM sys.objects WHERE type = 'P' AND name = 'GETFORMS')

BEGIN

DROP PROC GETFORMS

END

GO

create procedure GETFORMS

as

select \*

from TRANSPORTATION

INNER JOIN CLIENTS on TRANSPORTATION.IDCLIENTS = CLIENTS.IDCLIENTS

INNER JOIN TRANSPORTATION\_WATER\_LINK on TRANSPORTATION.idTRANSPORTATION = TRANSPORTATION\_WATER\_LINK.idTRANSPORTATION

INNER JOIN WATER on TRANSPORTATION\_WATER\_LINK.IDWATER = WATER.idWATER

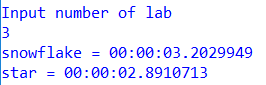
INNER JOIN TRANSPORTATION\_CAR\_LINK on TRANSPORTATION.idTRANSPORTATION = TRANSPORTATION\_CAR\_LINK.idTRANSPORTATION

INNER JOIN CAR on TRANSPORTATION\_CAR\_LINK.IDCAR = CAR.IDCAR

INNER JOIN ROUTE on TRANSPORTATION.IDROUTE = ROUTE.IDROUTE

INNER JOIN MANUFACTURER on MANUFACTURER.IDMANUFACTURER = CAR.IDMANUFACTURER

*Рис. 6. SQL-скрипт, що отримує дані зі сховища даних типу сніжинка після зведення.*



*Рис. 7. Результати виконання програм.*

**Висновок:**

В результаті виконання даної роботи було визначено, що запит, що виконується в сховищі даних типу сніжинка після зведення даних виконується довше, хоче вона більш нормалізована та детальна, внаслідок чого і втрачається швидкодія.