*Додаток 1*

Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з комп’ютерного практикуму № 1 з дисципліни

«Аналіз даних в інформаційних системах»

на тему: «Створення сховища даних»

Виконав студент ІП-13, Недельчев Євген Олександрович

(шифр, прізвище, ім'я, по батькові)

Перевірив Олійник Юрій Олександрович

( прізвище, ім'я, по батькові)

Київ 202 3

Комп’ютерний практикум 1

Тема – створення сховища даних.

Мета – ознайомитись з підходами до створення сховищ даних.

Для виконання даної лабораторної роботи було вибрано наступний набір даних: <https://www.kaggle.com/datasets/leminhnguyen/winter-olympics-1924-2018>

<https://www.kaggle.com/datasets/nitishsharma01/olympics-124-years-datasettill-2020?select=regions.csv>

Даний набір містить дані про зимові олімпійські ігри, які проходили з 1924 року до 2018. Для обробки візьмемо чотири початкових таблиці: discipline\_details, events\_medals, olympic\_events, regions.

Таблиця discipline\_details містить дані про всі дисципліни, які проходили на іграх, та про призерів в цих дисциплінах.

Таблиця events\_medals містить дані про перфоманс країн, які на іграх здобули хочу б одне призове місце.

Таблиця olympic\_events містить дані про власне самі олімпійські ігри. Де і коли проходили, скільки країн та атлетів брали участь, скільки було дисциплін та скільки медалей було розіграно.

Таблиця regions містить дані про країни та їхні кодування в міжнародному олімпійському комітеті (IOC).

Stage зона

Изображение выглядит как текст, памятная табличка, черный

Автоматически созданное описание

**Discipline\_details**

event\_number – номер олімпійських ігор

event\_year – рік проведення ігор

discipline – назва дисципліни

category – назва категорії

date – дата проведення

n\_participants – кількість учасників

n\_country\_participants – кількість країн-учасниць

gold\_medalist – ім’я золотого призера

gold\_country – назва країни призера

silver\_medalist – ім’я срібного призера

silver\_country – назва країни призера

bronze\_medalist – ім’я бронзового призера

bronze\_country – назва країни призера

**Olympic\_events**

event\_city – місто, в якому проводилися ігри

event\_country – країна, в якій проводилися ігри

event\_number – номер олімпійських ігор

event\_year – рік проведення ігор

opening\_ceremony – дата церемонії відкриття

closing\_ceremony – дата церемонії закриття

n\_participants – кількість атлетів-учасників

n\_countries - кількість країн-учасниць

n\_medals – кількість розіграних медалей

n\_disciplines – кількість дисциплін

**Events\_medals**

event\_number – номер олімпійських ігор

event\_year – рік проведення

country – назва країни

gold – кількість золотих медалей

silver – кількість срібних медалей

bronze – кількість бронзових медалей

total – сумарна кількість медалей

**Regions**

ioc – код країни

country – назва країни

Для створення stage зони було задіяно sql скрипт нижче:

USE winter\_olympics\_stage;  
  
DROP TABLE IF EXISTS discipline\_details;  
DROP TABLE IF EXISTS regions;  
DROP TABLE IF EXISTS events\_medals;  
DROP TABLE IF EXISTS olympic\_events;  
  
CREATE TABLE IF NOT EXISTS discipline\_details (  
 id INT NOT NULL AUTO\_INCREMENT,  
 event\_number VARCHAR(10),  
 event\_year YEAR,  
 discipline VARCHAR(255),  
 category VARCHAR(255),  
 `date` VARCHAR(30),  
 n\_participants INT,  
 n\_country\_participants INT,  
 gold\_medalist VARCHAR(255),  
 gold\_country VARCHAR(255),  
 silver\_medalist VARCHAR(255),  
 silver\_country VARCHAR(255),  
 bronze\_medalist VARCHAR(255),  
 bronze\_country VARCHAR(255),  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS events\_medals (  
 id INT NOT NULL AUTO\_INCREMENT,  
 event\_number VARCHAR(10),  
 event\_year YEAR,  
 country VARCHAR(255),  
 gold INT,  
 silver INT,  
 bronze INT,  
 total INT,  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS olympic\_events (  
 id INT NOT NULL AUTO\_INCREMENT,  
 event\_city VARCHAR(255),  
 event\_country VARCHAR(255),  
 event\_number VARCHAR(10),  
 event\_year YEAR,  
 opening\_ceremony VARCHAR(30),  
 closing\_ceremony VARCHAR(30),  
 n\_participants INT,  
 n\_countries INT,  
 n\_medals INT,  
 n\_disciplines INT,  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS regions (  
 id INT NOT NULL AUTO\_INCREMENT,  
 ioc VARCHAR(5),  
 country VARCHAR(255),  
 PRIMARY KEY (id)  
);

Скрипт для заповнення stage зони даними наведено нижче:

USE winter\_olympics\_stage;  
  
TRUNCATE TABLE discipline\_details;  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\discipline\_details.csv'  
 INTO TABLE discipline\_details  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (event\_number,  
 event\_year,  
 discipline,  
 category,  
 `date`,  
 n\_participants,  
 n\_country\_participants,  
 gold\_medalist,  
 gold\_country,  
 silver\_medalist,  
 silver\_country,  
 bronze\_medalist,  
 bronze\_country);  
  
TRUNCATE TABLE events\_medals;  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\events\_medals.csv'  
 INTO TABLE events\_medals  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
(event\_number,  
event\_year,  
country,  
gold,  
silver,  
bronze,  
total);  
  
TRUNCATE TABLE olympic\_events;  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\olympic\_events.csv'  
 INTO TABLE olympic\_events  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
( event\_city,  
 event\_country,  
 event\_number,  
 event\_year,  
 opening\_ceremony,  
 closing\_ceremony,  
 n\_participants,  
 n\_countries,  
 n\_medals,  
 n\_disciplines);  
  
TRUNCATE TABLE regions;  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\noc\_regions.csv'  
 INTO TABLE regions  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (ioc, country);  
  
# Remove newline symbols from the names of countries  
  
UPDATE regions SET country = *REPLACE*(*REPLACE*(country, '\r', ''), '\n', '');  
UPDATE discipline\_details SET bronze\_country = *REPLACE*(*REPLACE*(bronze\_country, '\r', ''), '\n', '');

Main сховище

Изображение выглядит как текст, черный, в помещении

Автоматически созданное описание

Фактові таблиці містять інформацію про перфоманс країн на іграх, про дисципліни та самі ігри, а також зовнішні ключі на таблиці-виміри, що местять детальні дані про країни, дисципліни, дати та місця проведення олімпійських ігор.

Скрипти для створення основного сховища наведені нижче:

USE winter\_olympics\_main;  
  
SET FOREIGN\_KEY\_CHECKS=0;  
DROP TABLE discipline\_fact;  
DROP TABLE athlete\_dim;  
DROP TABLE country\_dim;  
DROP TABLE country\_performance\_fact;  
DROP TABLE date\_dim;  
DROP TABLE discipline\_details\_dim;  
DROP TABLE event\_place\_dim;  
DROP TABLE olympic\_games\_fact;  
SET FOREIGN\_KEY\_CHECKS=1;  
  
CREATE TABLE IF NOT EXISTS discipline\_details\_dim (  
 id INT NOT NULL AUTO\_INCREMENT,  
 discipline VARCHAR(255),  
 category VARCHAR(255),  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS athlete\_dim (  
 id INT NOT NULL AUTO\_INCREMENT,  
 name VARCHAR(255),  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS country\_dim (  
 id INT NOT NULL AUTO\_INCREMENT,  
 country VARCHAR(255),  
 IOC VARCHAR(5),  
 source\_id int default null,  
 start\_date date default null,  
 end\_date date default null,  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS date\_dim (  
 id INT NOT NULL AUTO\_INCREMENT,  
 year YEAR,  
 PRIMARY KEY (id)  
);  
  
CREATE TABLE IF NOT EXISTS event\_place\_dim (  
 id INT NOT NULL AUTO\_INCREMENT,  
 country\_id INT,  
 city VARCHAR(80),  
 PRIMARY KEY (id),  
 FOREIGN KEY (country\_id) REFERENCES country\_dim (id)  
);  
  
CREATE TABLE IF NOT EXISTS olympic\_games\_fact (  
 id INT NOT NULL AUTO\_INCREMENT,  
 place\_id INT,  
 date\_id INT,  
 event\_number VARCHAR(10),  
 n\_participants INT,  
 n\_countries INT,  
 n\_disciplines INT,  
 n\_medals INT,  
 PRIMARY KEY (id),  
 FOREIGN KEY (place\_id) REFERENCES event\_place\_dim (id),  
 FOREIGN KEY (date\_id) REFERENCES date\_dim (id)  
);  
  
CREATE TABLE IF NOT EXISTS country\_performance\_fact (  
 id INT NOT NULL AUTO\_INCREMENT,  
 country\_id INT,  
 event\_id INT,  
 date\_id INT,  
 gold INT,  
 silver INT,  
 bronze INT,  
 total INT,  
 PRIMARY KEY (id),  
 FOREIGN KEY (country\_id) REFERENCES country\_dim (id),  
 FOREIGN KEY (event\_id) REFERENCES event\_place\_dim (id),  
 FOREIGN KEY (date\_id) REFERENCES date\_dim (id)  
);  
  
CREATE TABLE IF NOT EXISTS discipline\_fact (  
 id INT NOT NULL AUTO\_INCREMENT,  
 discipline\_details\_id INT,  
 event\_id INT,  
 gold\_medalist\_id INT,  
 gold\_country\_id INT,  
 silver\_medalist\_id INT,  
 silver\_country\_id INT,  
 bronze\_medalist\_id INT,  
 bronze\_country\_id INT,  
 PRIMARY KEY (id),  
 FOREIGN KEY (discipline\_details\_id) REFERENCES discipline\_details\_dim (id),  
 FOREIGN KEY (event\_id) REFERENCES olympic\_games\_fact(id),  
 FOREIGN KEY (gold\_medalist\_id) REFERENCES athlete\_dim(id),  
 FOREIGN KEY (gold\_country\_id) REFERENCES country\_dim(id),  
 FOREIGN KEY (silver\_medalist\_id) REFERENCES athlete\_dim(id),  
 FOREIGN KEY (silver\_country\_id) REFERENCES country\_dim(id),  
 FOREIGN KEY (bronze\_medalist\_id) REFERENCES athlete\_dim(id),  
 FOREIGN KEY (bronze\_country\_id) REFERENCES country\_dim(id)  
);

Скрипти для заповнення основного сховища та зв’язування даних між собою:

use winter\_olympics\_stage;  
  
INSERT INTO winter\_olympics\_main.discipline\_details\_dim (discipline, category)  
SELECT DISTINCT discipline, category  
FROM winter\_olympics\_stage.discipline\_details;  
  
INSERT INTO winter\_olympics\_main.athlete\_dim (name)  
SELECT DISTINCT medalist\_name  
FROM (SELECT gold\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details  
 UNION  
 SELECT silver\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details  
 UNION  
 SELECT bronze\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details) as medalists;  
  
INSERT INTO winter\_olympics\_main.country\_dim (country, IOC)  
SELECT country, ioc  
FROM winter\_olympics\_stage.regions;  
  
INSERT INTO winter\_olympics\_main.date\_dim (YEAR)  
SELECT DISTINCT `year`  
FROM (SELECT event\_year as `year`  
 FROM discipline\_details  
 UNION  
 SELECT event\_year as `year`  
 FROM events\_medals  
 UNION  
 SELECT event\_year as `year`  
 FROM olympic\_events) as years;  
  
INSERT INTO winter\_olympics\_main.event\_place\_dim (country\_id, city)  
SELECT DISTINCT winter\_olympics\_main.country\_dim.id, olympic\_events.event\_city  
FROM olympic\_events  
 JOIN winter\_olympics\_main.country\_dim  
 ON olympic\_events.event\_country = country\_dim.country;  
  
INSERT INTO winter\_olympics\_main.olympic\_games\_fact (place\_id, date\_id, event\_number, n\_participants, n\_countries,  
 n\_disciplines, n\_medals)  
SELECT ep.id AS place\_id,  
 dd.id AS date\_id,  
 o.event\_number,  
 o.n\_participants,  
 o.n\_countries,  
 o.n\_disciplines,  
 o.n\_medals  
FROM winter\_olympics\_stage.olympic\_events o  
 JOIN winter\_olympics\_main.event\_place\_dim ep ON o.event\_city = ep.city  
 JOIN winter\_olympics\_main.date\_dim dd ON o.event\_year = dd.year;  
  
SET FOREIGN\_KEY\_CHECKS = 0;  
INSERT INTO winter\_olympics\_main.country\_performance\_fact (country\_id, event\_id, date\_id, gold, silver, bronze, total)  
SELECT cd.id as country\_id, og.id as event\_id, dd.id as date\_id, em.gold, em.silver, em.bronze, em.total  
FROM events\_medals as em  
 JOIN winter\_olympics\_main.country\_dim cd on cd.country = em.country  
 JOIN winter\_olympics\_main.olympic\_games\_fact og on og.event\_number = em.event\_number  
 JOIN winter\_olympics\_main.date\_dim dd on dd.year = em.event\_year;  
SET FOREIGN\_KEY\_CHECKS = 1;  
  
INSERT INTO winter\_olympics\_main.discipline\_fact (discipline\_details\_id, event\_id, gold\_medalist\_id, gold\_country\_id,  
 silver\_medalist\_id, silver\_country\_id, bronze\_medalist\_id,  
 bronze\_country\_id)  
SELECT dd.id as discipline\_details\_id,  
 og.id as event\_id,  
 adg.id as gold\_medalist\_id,  
 cdg.id as gold\_country\_id,  
 ads.id as silver\_medalist\_id,  
 cds.id as silver\_country\_id,  
 adb.id as bronze\_medalist\_id,  
 cdb.id as bronze\_country\_id  
FROM discipline\_details as disciplines  
 JOIN winter\_olympics\_main.discipline\_details\_dim dd  
 ON (dd.category = disciplines.category AND dd.discipline = disciplines.discipline)  
 JOIN winter\_olympics\_main.olympic\_games\_fact og  
 ON og.event\_number = disciplines.event\_number  
 JOIN winter\_olympics\_main.athlete\_dim adg  
 ON adg.name = disciplines.gold\_medalist  
 JOIN winter\_olympics\_main.country\_dim as cdg  
 ON cdg.IOC = disciplines.gold\_country  
 JOIN winter\_olympics\_main.athlete\_dim ads  
 ON ads.name = disciplines.silver\_medalist  
 JOIN winter\_olympics\_main.country\_dim as cds  
 ON cds.IOC = disciplines.silver\_country  
 JOIN winter\_olympics\_main.athlete\_dim adb  
 ON adb.name = disciplines.bronze\_medalist  
 JOIN winter\_olympics\_main.country\_dim as cdb  
 ON cdb.IOC = disciplines.bronze\_country

Скрипт для реалізації можливості slowly changing dimension:

use winter\_olympics\_main;  
  
drop procedure if exists *slow\_change\_countries*;  
  
delimiter //  
create procedure *slow\_change\_countries*(old\_name varchar(255), old\_ioc varchar(5), new\_name varchar(255),  
 new\_ioc varchar(255))  
begin  
 declare old\_country\_id int default null;  
  
 select distinct id  
 into old\_country\_id  
 from country\_dim  
 where country = old\_name and ioc = old\_ioc;  
  
 if old\_name is null then  
 signal sqlstate '45000' set message\_text = 'Such country does not exist';  
 else  
 insert into country\_dim (country, IOC, source\_id, start\_date)  
 value (new\_name, new\_ioc, old\_country\_id, *CURRENT\_DATE*);  
  
 update country\_dim  
 set end\_date = *CURRENT\_DATE* where old\_country\_id = id;  
 end if;  
end //  
delimiter ;  
  
call *slow\_change\_countries*('Russia', 'RUS', 'Moskovia', 'MSK');

Скрипт для можливості завантаження нових даних до існуючих:

use winter\_olympics\_main;  
  
truncate table winter\_olympics\_stage.discipline\_details;  
truncate table winter\_olympics\_stage.events\_medals;  
truncate table winter\_olympics\_stage.regions;  
truncate table winter\_olympics\_stage.olympic\_events;  
  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\discipline\_details2.csv'  
 INTO TABLE winter\_olympics\_stage.discipline\_details  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (event\_number,  
 event\_year,  
 discipline,  
 category,  
 `date`,  
 n\_participants,  
 n\_country\_participants,  
 gold\_medalist,  
 gold\_country,  
 silver\_medalist,  
 silver\_country,  
 bronze\_medalist,  
 bronze\_country);  
  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\events\_medals2.csv'  
 INTO TABLE winter\_olympics\_stage.events\_medals  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (event\_number,  
 event\_year,  
 country,  
 gold,  
 silver,  
 bronze,  
 total);  
  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\olympic\_events2.csv'  
 INTO TABLE winter\_olympics\_stage.olympic\_events  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (event\_city,  
 event\_country,  
 event\_number,  
 event\_year,  
 opening\_ceremony,  
 closing\_ceremony,  
 n\_participants,  
 n\_countries,  
 n\_medals,  
 n\_disciplines);  
  
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\noc\_regions2.csv'  
 INTO TABLE winter\_olympics\_stage.regions  
 FIELDS TERMINATED BY ','  
 ENCLOSED BY '"'  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS  
 (ioc, country);  
  
  
  
INSERT INTO discipline\_details\_dim (discipline, category)  
SELECT DISTINCT discipline, category  
FROM winter\_olympics\_stage.discipline\_details dd  
WHERE NOT *EXISTS*(SELECT discipline, category  
 FROM discipline\_details\_dim ddd  
 WHERE dd.discipline = ddd.discipline  
 AND dd.category = ddd.category);  
  
INSERT INTO athlete\_dim (name)  
SELECT DISTINCT medalist\_name  
FROM (SELECT gold\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details  
 UNION  
 SELECT silver\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details  
 UNION  
 SELECT bronze\_medalist AS medalist\_name  
 FROM winter\_olympics\_stage.discipline\_details) as medalists  
WHERE NOT *EXISTS*(SELECT medalist\_name FROM athlete\_dim ad WHERE medalists.medalist\_name = ad.name);  
  
  
INSERT INTO country\_dim (country, IOC)  
SELECT DISTINCT country, ioc  
FROM winter\_olympics\_stage.regions r  
WHERE NOT *EXISTS*(SELECT country, IOC  
 FROM country\_dim cd  
 WHERE cd.IOC = r.ioc  
 AND cd.country = r.country);  
  
INSERT INTO date\_dim (YEAR)  
SELECT DISTINCT `year`  
FROM (SELECT event\_year as `year`  
 FROM winter\_olympics\_stage.discipline\_details  
 UNION  
 SELECT event\_year as `year`  
 FROM winter\_olympics\_stage.events\_medals  
 UNION  
 SELECT event\_year as `year`  
 FROM winter\_olympics\_stage.olympic\_events) as years  
WHERE NOT *EXISTS*(SELECT year FROM date\_dim dd WHERE dd.year = years.year);  
  
INSERT INTO event\_place\_dim (country\_id, city)  
SELECT DISTINCT cd.id, oe.event\_city  
FROM winter\_olympics\_stage.olympic\_events oe  
 JOIN country\_dim cd  
 ON oe.event\_country = cd.country  
WHERE NOT *EXISTS*(SELECT country\_id, city  
 FROM event\_place\_dim ed  
 WHERE oe.event\_city = ed.city);  
  
INSERT INTO olympic\_games\_fact (place\_id, date\_id, event\_number, n\_participants, n\_countries,  
 n\_disciplines, n\_medals)  
SELECT DISTINCT ep.id AS place\_id,  
 dd.id AS date\_id,  
 o.event\_number,  
 o.n\_participants,  
 o.n\_countries,  
 o.n\_disciplines,  
 o.n\_medals  
FROM winter\_olympics\_stage.olympic\_events o  
 JOIN event\_place\_dim ep ON o.event\_city = ep.city  
 JOIN date\_dim dd ON o.event\_year = dd.year  
WHERE NOT *EXISTS*(SELECT og.place\_id, og.date\_id, og.event\_number  
 FROM olympic\_games\_fact og  
 WHERE ep.id = og.place\_id  
 AND dd.id = og.date\_id  
 AND o.event\_number = og.event\_number);  
  
SET FOREIGN\_KEY\_CHECKS = 0;  
INSERT INTO country\_performance\_fact (country\_id, event\_id, date\_id, gold, silver, bronze, total)  
SELECT DISTINCT cd.id as country\_id, og.id as event\_id, dd.id as date\_id, em.gold, em.silver, em.bronze, em.total  
FROM winter\_olympics\_stage.events\_medals as em  
 JOIN country\_dim cd on cd.country = em.country  
 JOIN olympic\_games\_fact og on og.event\_number = em.event\_number  
 JOIN date\_dim dd on dd.year = em.event\_year  
WHERE NOT *EXISTS*(SELECT cp.country\_id, cp.event\_id, cp.date\_id  
 FROM country\_performance\_fact cp  
 WHERE cd.id = cp.country\_id  
 AND og.id = cp.id  
 AND dd.id = cp.date\_id);  
SET FOREIGN\_KEY\_CHECKS = 1;  
  
INSERT INTO discipline\_fact (discipline\_details\_id, event\_id, gold\_medalist\_id, gold\_country\_id,  
 silver\_medalist\_id, silver\_country\_id, bronze\_medalist\_id,  
 bronze\_country\_id)  
SELECT DISTINCT dd.id as discipline\_details\_id,  
 og.id as event\_id,  
 adg.id as gold\_medalist\_id,  
 cdg.id as gold\_country\_id,  
 ads.id as silver\_medalist\_id,  
 cds.id as silver\_country\_id,  
 adb.id as bronze\_medalist\_id,  
 cdb.id as bronze\_country\_id  
FROM winter\_olympics\_stage.discipline\_details as disciplines  
 JOIN discipline\_details\_dim dd  
 ON (dd.category = disciplines.category AND dd.discipline = disciplines.discipline)  
 JOIN olympic\_games\_fact og  
 ON og.event\_number = disciplines.event\_number  
 JOIN athlete\_dim adg  
 ON adg.name = disciplines.gold\_medalist  
 JOIN country\_dim as cdg  
 ON cdg.IOC = disciplines.gold\_country  
 JOIN athlete\_dim ads  
 ON ads.name = disciplines.silver\_medalist  
 JOIN country\_dim as cds  
 ON cds.IOC = disciplines.silver\_country  
 JOIN athlete\_dim adb  
 ON adb.name = disciplines.bronze\_medalist  
 JOIN country\_dim as cdb  
 ON cdb.IOC = disciplines.bronze\_country  
WHERE NOT *EXISTS*(SELECT df.event\_id, df.discipline\_details\_id  
 FROM discipline\_fact df  
 WHERE og.id = df.event\_id  
 AND dd.id = df.discipline\_details\_id);

Висновок

У цьому комп’ютерному практикуму ми ознайомились з можливістю проектування сховища даних, проходячи етапи створення stage зони для завантаження даних та створення основного сховища для розподілення даних за певною логікою зі зв’язками між ними. Було реалізовано можливості slowly changing dimension та завантаження нових даних до вже існуючих. Попередньо дані були оброблені, щоб привести їх до потрібного формату.