МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КПІ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА АСОІУ

Звіт

з дисципліни

«Організація баз даних та знань»

Лабараторна робота №3

Виконав :

Студент групи ІС-32

№Залікової книжки - 3407

Касянчик Д.О.

Перевірила :

Томахіна О. С.

Київ – 2015

1. Мета роботи

Опанувати створення запитів мовою SQL з використанням під запитів та з’єднань.

1. Опис предметної області

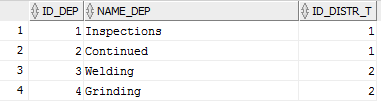
База конструкторсько-технологічної інформації підприємства зберігає наступну + ієрархічній специфікації (Bill of Material, BOM). Складовими BOM є деталі або зборочні одиниці (ДЗО). Спеціфікація продукту, як і специфікації його складових (зборочних одиниць, тобто агрегатів), містить перелік деталей або зборочних одиниць (ДЗО), які входять у даний продукт або агрегат. ДЗО або закупаються, або виробляються. Кожна ДЗО має вагу і собівартість.

* Якщо ДЗО виробляється на підприємстві, Відділ Головного технолога визначає для неї *технологічний маршрут* по певних дільницях певних цехів, починаючи з тої дільниці, де вона виникає вперше, і закінчуючи операцією зборки, згідно якій вона потрапляє в зборочну одиницю вищого рівня.
* На кожній дільниці над ДЗО, яка там обробляється, виконуються певні *технологічні операції* певної тривалості, яка залежить від ДЗО. Кожна технологічна операція має вартість одиниці часу (хвилини), куди входять вартість часу роботи обладнання і вартість часу роботи робітника, що обслуговує це обладнання.

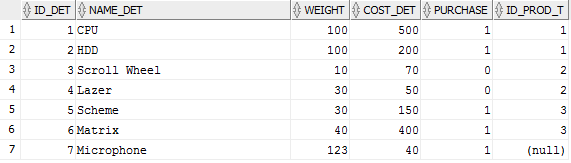
По цій базі менеджери підприємства аналізують планову собівартість виробництва продуктів.

1. Таблиці використанні в вправах

DEPARTMENT



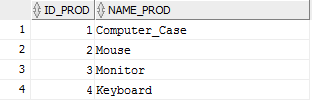
DETAILS



DISTRICT

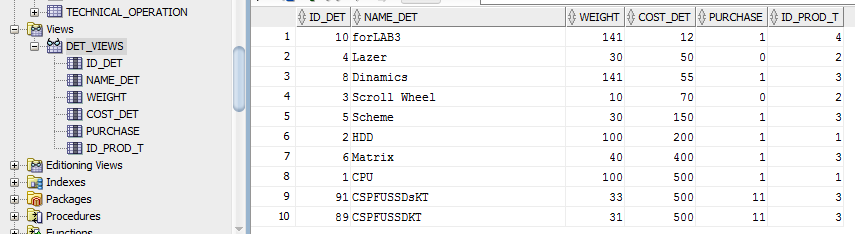


PRODUCT



1. Вправи
2. Вправа

Створення подання, яке виводить усі елементи довідника в порядку кодів.



Код : CREATE VIEW DET\_VIEWS AS SELECT \* FROM DETAILS ORDER BY COST\_DET;

Як ми бачимо по Screen Shot’i були виведені всі дані відсортовані по вартості

1. Вправа

Створення запиту на вибірку з логічними операціями в умовах відбору записів.



Код :

SELECT \* FROM DETAILS WHERE COST\_DET > 200 AND PURCHASE = 1 AND WEIGHT >50;

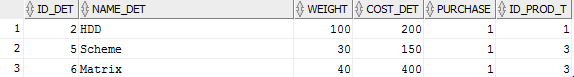
Як ми бачимо по Screen Shot’i були виведені дані які закупляються та вартість яких більше 200 і вага більше 50

1. Вправа

У базі індивідуальної предметної області, користуючись підзапитами, в т.ч.конструкціями EXISTS та IN, реалізувати у SQL три запити до двох і більше таблиць.



Screen Shot «EXIST»



Screen Shot «IN»

Код для EXIST :

SELECT \* FROM PRODUCT WHERE EXISTS (SELECT \* FROM DETAILS WHERE ID\_PROD\_T >= '2' AND ID\_PROD = ID\_PROD\_T );

Код для IN :

SELECT \* FROM DETAILS WHERE ID\_DET IN (SELECT ID\_DET\_TECH FROM OPERATION\_DETAILS);

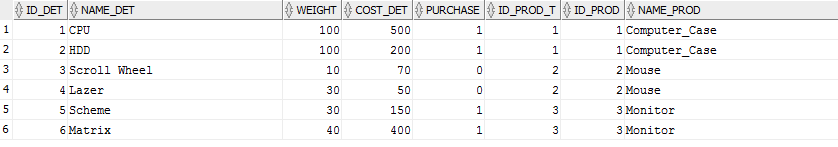
Як ми бачимо по Screen Shot’i «EXIST» були виведені правильні дані, тобто лише ті, якщо вони існують, де ID більше 2.

Як ми бачимо по Screen Shot’i «IN» були виведені правильні дані, тобто лише ті, якщо вони існують, де ID двох таблиць співпадають.

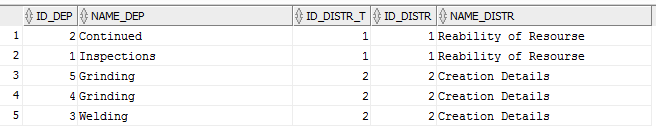
1. Вправа

Для індивідуальної предметної області, користуючись конструкціями JOIN та іншими з’єднаннями, реалізувати у SQL наступні запити:

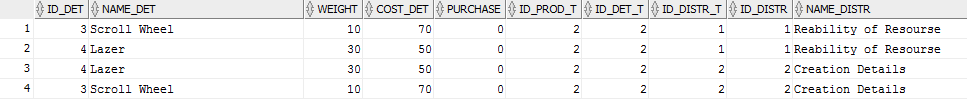
* Запит з внутрішнім з’єднанням таблиць;
* Запит з лівим з’єднанням таблиць;
* Запит з правим з’єднанням таблиць.



Screen Shot “INNER JOIN”



Screen Shot “LEFT JOIN”



Screen Shot “Right JOIN”

Код для INNER JOIN :

SELECT \* FROM DETAILS

INNER JOIN PRODUCT

ON DETAILS.ID\_PROD\_T =PRODUCT.ID\_PROD;

Код для LEFT JOIN :

SELECT \* FROM DEPARTMENT

LEFT JOIN DISTRICT

ON ID\_DISTR\_T=ID\_DISTR;

Код для RIGHT JOIN ( з використанням LEFT JOIN) :

SELECT \* FROM DETAILS

LEFT JOIN DISTRICT\_DETAILS

ON DETAILS.ID\_PROD\_T = DISTRICT\_DETAILS.ID\_DET\_T

RIGHT JOIN DISTRICT

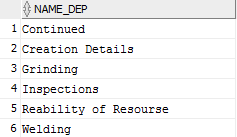
ON DISTRICT\_DETAILS.ID\_DISTR\_T = DISTRICT.ID\_DISTR;

Як ми бачимо по Screen Shot’i «INNER JOIN» були виведені правильні дані, тобто лише ті, якщо співпадають для двох таблиць.

Як ми бачимо по Screen Shot’i «LEFT JOIN» були виведені правильні дані, тобто була взята ціла ліва таблиця, якщо знаходився відповідний їй запис, то записувалися певні дані, в іншому ж випадку на їхньому місці записувався NULL. Для «RIGHT JOIN» все з точністю до навпаки.

1. Вправа

Для індивідуальної предметної області, користуючись конструкцією UNION, реалізувати у SQL запит на об’єднання таблиць.



Код для UNION :

SELECT NAME\_DEP FROM DEPARTMENT

UNION

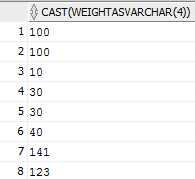
SELECT NAME\_DISTR FROM DISTRICT;

Як ми бачимо по Screen Shot’і були виведені всі назви, об’єднані з двох таблиць.

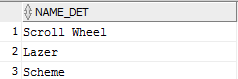
1. Вправа

Для індивідуальної предметної області, користуючись необхідними конструкціями, інформацію про які знайти самостійно, реалізувати наступні запити:

* з перетворенням типу даних результату і запиту (наприклад, дату в рядок, рядок в число, тощо)
* з пошуком фрагменту текстового поля (наприклад, знайти всі прізвища, що закінчуються на «ко»)
* з вибіркою та обробкою порожніх значень (NULL), заміною їх на інші з використанням умови



Screen Shot “CAST”



Screen Shot “LIKE”



Screen Shot “NULL\_WORK\_UPDATE”

Код для CAST :

SELECT CAST(WEIGHT AS varchar(4)) from DETAILS;

Код для LIKE :

SELECT NAME\_DET FROM DETAILS

WHERE NAME\_DET LIKE '%er%'

OR NAME\_DET LIKE 'Sc%';

Код для “NULL\_WORK\_UPDATE” :

UPDATE DETAILS SET COST\_DET = '0' WHERE COST\_DET IS NULL;

Як ми бачимо по Screen Shot’i «CAST» зміна, яка мала тип int була трансформована в CHAR.

Як ми бачимо по Screen Shot’i «LIKE» було знайдено таку деталі, яка містила в собі частинку «er» або починалися «Sc» на .

Як ми бачимо по Screen Shot’i «NULL\_WORK\_UPDATE» було знайдено такі поля в стовбці які були NULL та замінили їх на число ‘0’.

1. Висновок

В процесі виконання лабораторної роботи ознайомився з основними запитами мови SQL. Зрозумів, яким чином працюють такі оператори як INNER JOIN, LEFT JOIN, RIGHT JOIN , зрозумів різницю між ними . Також зрозумів, яким чином ми здійснюємо пошук в словах якоїсь частинки (тобто оператор LIKE). Опанував робота над змінними, які мають значення NULL. Навчився перетворювати колонки одного типу в інший .

1. Код реалізації БД

CREATE TABLE CITIZENS

(

ID\_TCitizen int PRIMARY KEY,

Name char(30) not null,

Surname char(30) not null,

Birthday date not null,

ID\_Statement int,

ID\_Passport int ,

ID\_PassportTable int,

FOREIGN KEY (ID\_Statement) REFERENCES STATEMENT(ID\_TStatement),

FOREIGN KEY (ID\_Passport) REFERENCES PASSPORT(ID\_TPassport),

FOREIGN KEY (ID\_PassportTable) REFERENCES PASSPORTTABLE(ID\_TPassportTable)

);

CREATE TABLE STATEMENT

(

ID\_TStatement int PRIMARY KEY,

Type char(30),

ID\_Document int,

Availability int,

FOREIGN KEY (ID\_Document) REFERENCES DOCUMENTS(ID\_TDocument)

);

CREATE TABLE DOCUMENTS

(

ID\_TDocument int PRIMARY KEY,

DocName char(30) not null

);

CREATE TABLE PASSPORT

(

ID\_TPassport int PRIMARY KEY,

SERIES char(20) not null,

OutputDate date not null,

IssuedByPT int not null,

ID\_Operation int ,

FOREIGN KEY (ID\_Operation) REFERENCES ADRESSOPPS(ID\_TAdressOPP)

);

CREATE TABLE ADRESSOPPS

(

ID\_TAdressOPP int PRIMARY KEY,

Type char(20),

Adress char(20)

);

CREATE TABLE PASSPORTTABLE

(

ID\_TPassportTable int PRIMARY KEY,

Adress char(20) not null,

ID\_Citizen int ,

ID\_Police int,

FOREIGN KEY (ID\_Police) REFERENCES POLICE(ID\_TPolice)

);

CREATE TABLE POLICE

(

ID\_TPolice int PRIMARY KEY,

Adress char(20) not null,

ID\_Cause int ,

FOREIGN KEY (ID\_Cause) REFERENCES CAUSE(ID\_TCause)

);

CREATE TABLE CAUSE

(

ID\_TCause int PRIMARY KEY,

Surname char(30) not null,

Series char(20) not null,

CauseDate date not null

);

ALTER TABLE PASSPORTTABLE ADD FOREIGN KEY (ID\_Citizen) REFERENCES CITIZENS(ID\_TCitizen);

ALTER TABLE PASSPORT MODIFY ISSUEDBYPT char(30);

ALTER TABLE STATEMENT MODIFY Availability char(2);

ALTER TABLE STATEMENT ADD StateID int;

ALTER TABLE STATEMENT MODIFY StateID int not null;

DELETE FROM PASSPORT WHERE ID\_TPASSPORT=1;

DELETE FROM DOCUMENTS WHERE ID\_TDocument=1;

DELETE FROM STATEMENT WHERE ID\_TStatement >= 1;

INSERT INTO ADRESSOPPS VALUES ('1', 'Extract' , '40.y of Victory');

INSERT INTO ADRESSOPPS VALUES ('2', 'Input', 'st.Naberezhna 28');

INSERT INTO ADRESSOPPS VALUES ('3', 'Extract' , 'st. Baraky 12');

INSERT INTO ADRESSOPPS VALUES ('4', 'Input' , 'st. Stepana Bandery');

INSERT INTO ADRESSOPPS VALUES ('5', 'Extract' , 'st. Shuhevycha');

INSERT INTO PASSPORT VALUES ('1' , 'GF3256', '21.02.2014', 'Rivne RV UDMS' , '1');

INSERT INTO PASSPORT VALUES ('2' , 'DE3679', '10.07.2014', 'Dubno RV UDMS' , '2');

INSERT INTO PASSPORT VALUES ('3' , 'BM1569', '01.10.2014', 'Donezk RV UDMS' , '3');

INSERT INTO PASSPORT VALUES ('4' , 'TG3232', '07.12.2014', 'Lviv RV UDMS' , '4');

INSERT INTO PASSPORT VALUES ('5' , 'ER3258', '18.06.2014', 'Kyiv RV UDMS' , '5');

INSERT INTO DOCUMENTS VALUES ('1', 'Birthday certificate');

INSERT INTO DOCUMENTS VALUES ('2', 'Extract certificate ');

INSERT INTO DOCUMENTS VALUES ('3', 'Photo 3x4');

INSERT INTO DOCUMENTS VALUES ('4', 'House office certificate');

INSERT INTO DOCUMENTS VALUES ('5', 'Consent for registration');

INSERT INTO CAUSE VALUES ('1' , 'Ivanovv', 'GF3256', '21.03.2014');

INSERT INTO CAUSE VALUES ('2' , 'Hatka', 'DM2186', '04.04.2014');

INSERT INTO STATEMENT VALUES ('1' , 'Lost passport' , '1', '+', '1');

INSERT INTO STATEMENT VALUES ('2' , 'Lost passport' , '3', '+', '1');

INSERT INTO STATEMENT VALUES ('3' , 'Lost passport' , '4', '+', '1');

INSERT INTO STATEMENT VALUES ('4' , 'Found passport', '1', '+', '2');

INSERT INTO STATEMENT VALUES ('5' , 'Found passport', '3', '-', '2');

INSERT INTO STATEMENT VALUES ('6' , 'First passport', '1', '+', '3');

INSERT INTO STATEMENT VALUES ('7' , 'First passport', '3', '+', '3');

INSERT INTO STATEMENT VALUES ('8' , 'First passport', '4', '+', '3');

INSERT INTO STATEMENT VALUES ('9' , 'First passport', '5', '+', '3');

INSERT INTO POLICE VALUES ('1', 'st.Grushevych', '1');

INSERT INTO POLICE VALUES ('2', 'st.Keruga 11', '2');

INSERT INTO PASSPORTTABLE VALUES ('1', 'st.Grushevych' , '1', '1');

INSERT INTO PASSPORTTABLE VALUES ('2', 'st.Grushevych' , '', '1');

INSERT INTO PASSPORTTABLE VALUES ('3', 'st.Grushevych' , '', '2');

INSERT INTO PASSPORTTABLE VALUES ('4', 'st.Keruga 11' , '', '1');

INSERT INTO CITIZENS VALUES ('2' , 'Masha', 'Alieca', '12.02.1996', '1' , '2' , '1');

INSERT INTO CITIZENS VALUES ('3' , 'Dima', 'Afert', '22.12.1994', '2' , '3' , '1');

INSERT INTO CITIZENS VALUES ('4' , 'Olena', 'Beru', '29.11.1994', '3' , '4' , '2');

INSERT INTO CITIZENS VALUES ('5' , 'Person', '1', '29.11.2002', '' , '' , '');

INSERT INTO CITIZENS VALUES ('6' , 'Person', '2', '29.11.2003', '' , '' , '');

------- TASK #1

SELECT \* FROM CITIZENS order by birthday;

------- TASK #2

SELECT \* FROM CITIZENS WHERE BIRTHDAY >= '01.01.93' AND BIRTHDAY < '01.01.2000';

------- TASK #3

SELECT TYPE, ID\_DOCUMENT FROM STATEMENT

WHERE EXISTS (SELECT ID\_TDocument FROM DOCUMENTS WHERE ID\_DOCUMENT > '1');

SELECT \* FROM CITIZENS

WHERE ID\_TCitizen IN (1,2);

------- TASK #4

SELECT NAME, SURNAME, Adress, ID\_PASSPORTTABLE , ID\_TPASSPORTTABLE FROM CITIZENS

INNER JOIN PASSPORTTABLE

ON CITIZENS.ID\_PASSPORTTABLE = PASSPORTTABLE.ID\_TPASSPORTTABLE;

SELECT NAME, SURNAME, Adress, ID\_PASSPORTTABLE , ID\_TPASSPORTTABLE FROM CITIZENS

FULL OUTER JOIN PASSPORTTABLE

ON CITIZENS.ID\_PASSPORTTABLE = PASSPORTTABLE.ID\_TPASSPORTTABLE

WHERE CITIZENS.ID\_PASSPORTTABLE IS NULL

OR PASSPORTTABLE.ID\_TPASSPORTTABLE IS NULL;

SELECT \* FROM CITIZENS

LEFT JOIN PASSPORTTABLE

ON CITIZENS.ID\_PASSPORTTABLE = PASSPORTTABLE.ID\_TPASSPORTTABLE;

SELECT \* FROM CITIZENS

RIGHT JOIN PASSPORTTABLE

ON CITIZENS.ID\_PASSPORTTABLE = PASSPORTTABLE.ID\_TPASSPORTTABLE;

------- TASK #5

SELECT SURNAME FROM CITIZENS

UNION

SELECT SURNAME FROM CAUSE;

------- TASK #6

SELECT SURNAME FROM CITIZENS

WHERE SURNAME LIKE '%ov%';

SELECT \* FROM CITIZENS

WHERE ID\_STATEMENT IS NULL;

UPDATE ADRESSOPPS SET TYPE = '0' WHERE TYPE IS NULL;

SELECT CAST(BIRTHDAY AS varchar(12)) from CITIZENS;