**ITS 26000 – Applied Database Tech**

**Lab 10 – (100 points)**

**Reload the tables again (Run the database scripts) before doing this exercise.**

**Use SQL to complete the following exercises.**

### **KimTay Pet Supplies:**

1. Create a view named MAJOR\_CUSTOMER. It consists of the customer ID, first name, last name, balance, credit limit, and rep ID for every customer whose credit limit is $500 or less. (25 points)
   1. Write and execute the CREATE VIEW command to create the MAJOR\_CUSTOMER view.

텍스트이(가) 표시된 사진

자동 생성된 설명

create view major\_customer as

select cust\_id,

first\_name,

last\_name,

balance,

credit\_limit,

rep\_id

from customer

where credit\_limit <= 500;

테이블이(가) 표시된 사진

자동 생성된 설명

* 1. Write and execute the command to retrieve the customer ID, first name, and last name of each customer in the MAJOR\_CUSTOMER view with a balance that exceeds the credit limit.

텍스트이(가) 표시된 사진

자동 생성된 설명

select cust\_id, first\_name, last\_name

from major\_customer

where balance > credit\_limit;

* 1. Write and execute the query that the DBMS actually executes.

select cust\_id, first\_name, last\_name

from (select cust\_id,

first\_name,

last\_name,

balance,

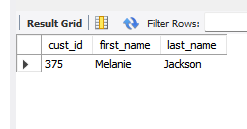
credit\_limit,

rep\_id

from customer

where credit\_limit <= 500) as A

where balance > credit\_limit;



* 1. Does updating the database through this view create any problems? If so, what are they? If not, why not? (10 points)

No, it will not cause any problems because it doesn’t have any aggregate functions, joins or group by statements

1. Create a view named ITEM\_INVOICE. It consists of the item ID, description, price, invoice number, invoice date, number ordered, and quoted price for all invoice lines currently on file. (25 points)
   1. Write and execute the CREATE VIEW command to create the ITEM\_INVOICE view

Create view item\_invoice as

select item.item\_id,

item.description,

price,

invoices.invoice\_num,

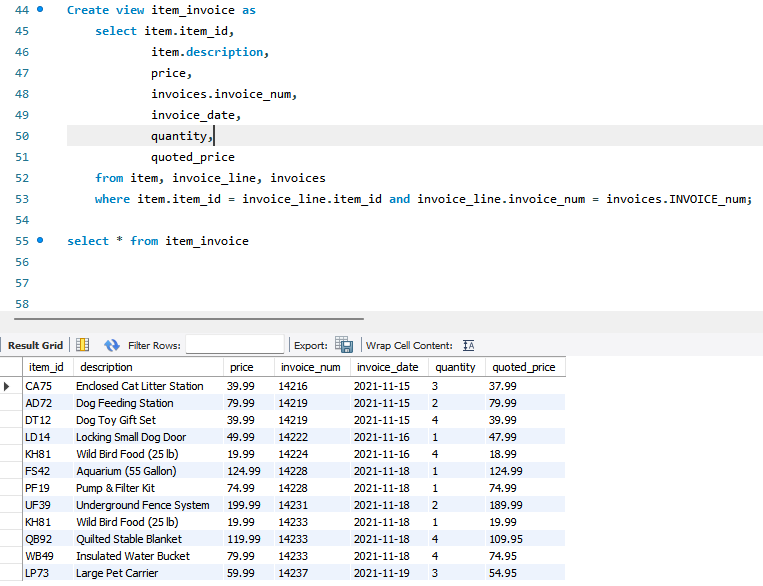
invoice\_date,

quantity,

quoted\_price

from item, invoice\_line, invoices

where item.item\_id = invoice\_line.item\_id and invoice\_line.invoice\_num = invoices.INVOICE\_num;



* 1. Write and execute the command to retrieve the item ID, description, invoice number, and quoted price for all invoices in the ITEM\_INVOICE view for items with quoted prices that exceed $100.

텍스트이(가) 표시된 사진

자동 생성된 설명

select item\_id, description, invoice\_num, quoted\_price

from item\_invoice

where quoted\_price > 100;

* 1. Write and execute the query that the DBMS actually executes.

select item\_id, description, invoice\_num, quoted\_price

from (select item.item\_id,

item.description,

price,

invoices.invoice\_num,

invoice\_date,

quantity,

quoted\_price

from item, invoice\_line, invoices

where item.item\_id = invoice\_line.item\_id and invoice\_line.invoice\_num = invoices.INVOICE\_num) as A

where quoted\_price > 100;

텍스트이(가) 표시된 사진

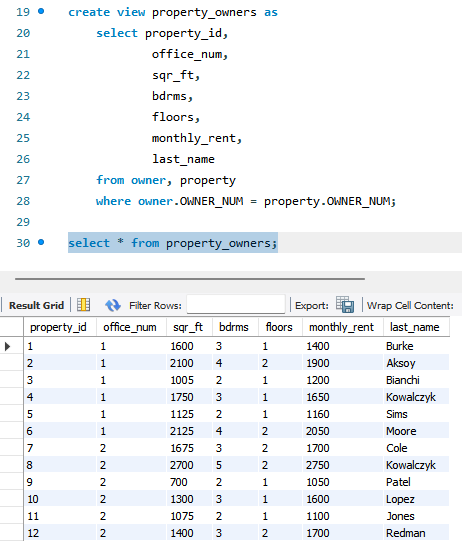
자동 생성된 설명

* 1. Does updating the database through this view create any problems? If so, what are they? If not, why not?

It depends on what you are updating. If you are updating information that is in only one table then it does not create any problems, however if you want to update information in multiple tables it will create problems.

### **StayWell Student Accommodation:**

1. Create a view named PROPERTY\_OWNERS. It consists of the property ID, office number, square footage, bedrooms, floors, monthly rent, and owner’s last name for every property in which the number of bedrooms is three. (25 points)
   1. Write and execute the CREATE VIEW command to create the PROPERTY\_OWNERS view.



create view property\_owners as

select property\_id,

office\_num,

sqr\_ft,

bdrms,

floors,

monthly\_rent,

last\_name

from owner, property

where owner.OWNER\_NUM = property.OWNER\_NUM;

* 1. Write and execute the command to retrieve the property ID, office number, monthly rent, square footage, and owner’s last name for every property in the PROPERTY\_OWNERS view with a monthly rent of less than $1675.

테이블이(가) 표시된 사진

자동 생성된 설명

select property\_id, office\_num, monthly\_rent, sqr\_ft, last\_name

from property\_owners

where monthly\_rent < 1675;

* 1. Write and execute the query that the DBMS actually executes.

테이블이(가) 표시된 사진

자동 생성된 설명

select property\_id, office\_num, monthly\_rent, sqr\_ft, last\_name

from (select property\_id,

office\_num,

sqr\_ft,

bdrms,

floors,

monthly\_rent,

last\_name

from owner, property

where owner.OWNER\_NUM = property.OWNER\_NUM) as A

where monthly\_rent < 1675;

* 1. Does updating the database through this view create any problems? If so, what are they? If not, why not?

It depends on what you are updating. If you are updating information that is in only one table then it does not create any problems, however if you want to update information in multiple tables it will create problems.

1. Create a view named MONTHLY\_RENTS. It consists of two columns: The first is the number of bedrooms, and the second is the average monthly rent for all properties in the PROPERTY table that have that number of bedrooms. Use AVERAGE\_RENT as the column name for the average monthly rent. Group and order the rows by number of bedrooms. (25 points)
   1. Write and execute the CREATE VIEW command to create the MONTHLY\_RENTS view.

텍스트이(가) 표시된 사진

자동 생성된 설명

create view monthly\_rents as

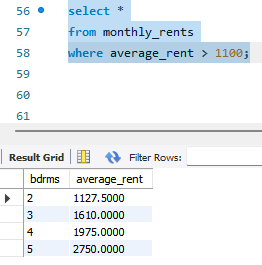
select bdrms, avg(monthly\_rent) as average\_rent

from property

group by bdrms

order by bdrms;

* 1. Write and execute the command to retrieve the bedrooms and average fee for each bedrooms for which the average fee is greater than $1,100.



select \*

from monthly\_rents

where average\_rent > 1100;

* 1. Write and execute the query that the DBMS actually executes.

텍스트이(가) 표시된 사진

자동 생성된 설명

select \*

from (select bdrms, avg(monthly\_rent) as average\_rent

from property

group by bdrms

order by bdrms) as a

where average\_rent > 1100;

* 1. Does updating the database through this view create any problems? If so, what are they? If not, why not?

Yes, it does create problems because it has the group by function.

**Deliverables**: **Document Report with the SQL statement/Query and snapshot of query, its execution and results.** Either word or pdf file need to be uploaded into Brightspace in one submission