Purpose - In this lab you will learn how to :

1. ALTER a VIEW
2. Distinguish the options for a column. Not Null With Default, Not Null, a different default than the natural one for data type or nothing specified.
3. Use information from physical design documents to implement tables
4. Grant Permissions giving other users access to your database. This is often the responsibility of the Database Administrator (DBA) so consider yourself the DBA of your database!

**Submission instruction: take frequent screenshot, write down the answers, mainly sql statements of the questions only as you proceed with this lab. There may be hand in instruction at the end of each lab.**

**How to Submit: Demonstration in Class only (check the deadline from course agenda and policies, late penalty 100%)**

1. In the previous lab, you created a VIEW called ORDER\_INFO and put it in your library BA210???. In that view you referred to a PART\_NUMBER and you did not know the actual part name.

Delete your ORDER\_INFO view and recreate it so you can include the part description. Note that part\_description is located in part table and you gave to join the part table appropriately with orders and oderline.

Try this

create view libraryname.order\_info as (

selecto.order\_number, o.order\_date, o.customer\_number, ol.part\_numb

er, ol.number\_ordered, ol.quoted\_price,(ol.number\_ordered\*ol.quoted\_

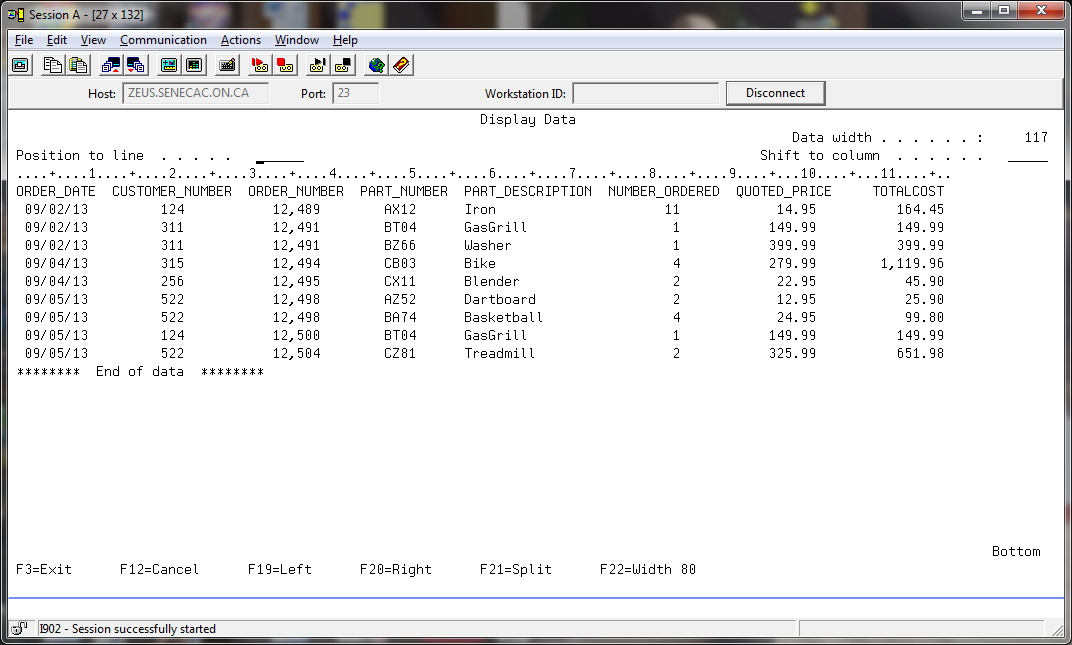
price) as TotalCOST, pr.part\_description

frompremiere.orderso,premiere.partpr,

premiere.orderlineol where

o.order\_number = ol.order\_number and ol.part\_number = pr.part\_numbe

r)



Unfortunately, a VIEW cannot be changed once it has been created. So to add the part description in our view we go through the following steps:

1. DROP the VIEW
2. Bring back the SQL command that created theVIEW
3. Make the changes to add the part description to the VIEWand
4. CREATE the VIEW again

Perform the following SQL commands:

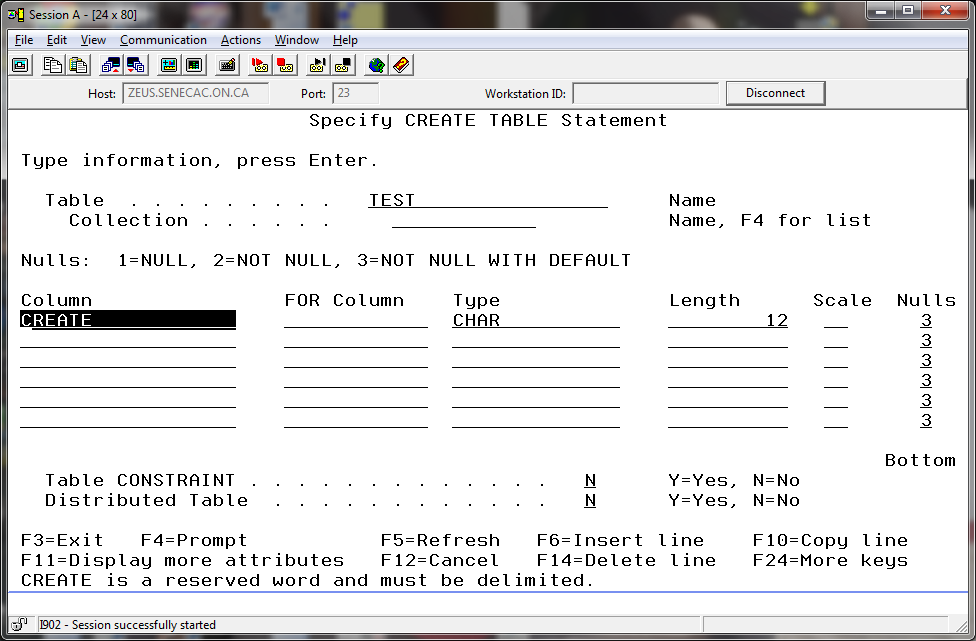
===>DROP VIEW BS201???.ORDER\_INFO .

Now, using F9 a few times or paging up and searching for the statement to press F9 on, bring back the original CREATE VIEW command and change it by including the PART table and the part description column.

This information can be found in the PART table. Unfortunately the table name “PART” is a reserved word. When you refer to any reserved word in SQL where it represents something else, you need to encase your references to it in double quotes.

“PART”

Here an attempt was made to use CREATE as a column name.



This was fixed by putting double quotes around the column name “CREATE”

The statement below was not accepted when prompting. So the reserved word “CREATE” had to include double quotes.

CREATE TABLE TEST("CREATE" CHAR ( 12) NOT NULL WITH DEFAULT)

It does run without the double quotes when not prompting.

CREATE TABLE TEST(CREATE CHAR ( 12) NOT NULL WITH DEFAULT

Press enter to execute the create view command you have constructed to include the part description. Run a Select statement to VIEW the data. If you are using Client Access with a setting of 80 columns, you may not be able to see the TOTAL\_COST column. **Use F20 to shift to the right side of the report and F19 to shift back to the left side of the report.**

Note that if your report is very long, then you have probably forgotten to use the WHERE clause to match the rows of the two tables!! In that case you would have produced the dreaded CARTESIAN JOIN (every row of table 1 joined to every row of table 2).

**2.The three options for a column or field – NOT NULL WITH DEFAULT, NOT NULL, a different default than the natural one for a data type or nothing specified.**

You can bypass this section if you know the differences with the four possible choices above.

But ... you can also run it and prove it works as advertised.

If you were wondering about the options for a column, you could devise a little experiment that shows what happens with the four choices.

We have AONE, ATWO, ATHREE and AFOUR as column names for character fields. The main difference is choosing not null with default, not null, not adding a clause, or specifying a different default that the usual one for that column type after defining the type and size CHAR (5)

CREATE TABLE TESTING( AONE CHAR (5) NOT NULL WITH DEFAULT,

ATWO CHAR (5) NOT NULL,

ATHREE CHAR (5),

AFOUR CHAR(5) WITH DEFAULT ‘XXXXX’ )

**Lets just put something in AONE and ignore ATWO, ATHREE and AFOUR**

INSERT INTO BA201B40/TESTING (AONE) VALUES('A')

Null values not allowed in column or variable ATWO.

**It did not work.**

**Lets just put something in ATHREE and ignore AONE ATWO and AFOUR**

INSERT INTO BA201B40/TESTING (ATHREE) VALUES('A')

Null values not allowed in column or variable ATWO

**It did not work.**

**Lets just put something in ATWO and ignore AONE, ATHREE and AFOUR**

INSERT INTO BA201B40/TESTING (ATWO) VALUES('A')

1 rows inserted in TESTING in BA201B40

**It worked. It is only necessary to supply a value for ATWO**.

**What was put in AONE, ATHREE and AFOUR?**

SELECT \* FROM TESTING

AONE ATWO ATHREE AFOUR

A - XXXXX

\*\*\*\*\*\*\*\* End of data \*\*\*\*\*\*\*\*

**AONE has spaces, ATWO has the letter ‘A’ ,ATHREE is showing '-' representing null and AFOUR has our specified default value.**

**Can you refer to null with the minus sign?**

DELETE FROM TESTING WHERE ATHREE = '-'

Row not found for DELETE

**No, you need to specify NULL.**

DELETE FROM TESTING WHERE ATHREE IS NULL

1 rows deleted from TESTING in BA201B40

You may want to drop the table and try three numeric fields and three date fields to see what the default values are.

When you say Not Null with default – what is the default for a date column when you do not specify a value when inserting a row?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_current date?

**3. Using information from physical design documents to implement tables**

Create the following two tables in your PRMIERxxx collection. (Don’t forget the constraints!)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Column | Data  Type | Length | PK | FK | NOT  NULL | Unique | Check | Default |
| Table: STUDENT |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| Campus\_no | | Char | 1 | Y |  |  |  |  |  |
| Stud\_no | | Numeric | 3 | Y |  |  |  |  |  |
| Stud\_lname | | Char | 15 |  |  |  |  |  |  |
| Stud\_fname | | Char | 15 |  |  |  |  |  |  |
| Stud\_pgm | | Char | 3 |  | PGM(Pgm\_id) |  |  |  | ‘CPA’ |
| Locker\_no | | Numeric | 3 |  |  |  | Y |  |  |
|  | |  |  |  |  |  |  |  |  |
| Table: PGM | |  |  |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |  |  |
| Pgm\_id | | Char | 3 | Y |  |  |  | ‘CPA’, ‘CPD’ |  |
| Pgm\_name | | Char | 40 |  |  | Y | Y |  |  |
| No\_semesters | | Numeric | 1 |  |  | Y |  | > = 0 | 6 |

Note: a 2-Part primary key constraint must be defined after the fields are defined, usually at the bottom of the CREATE TABLE command.

Ex.

create table PGM (Pgm\_id char(3),Pgm\_name char(40) not null, No\_seme

sters decimal(1) with default 6,

constraintPgm\_pk primary key (Pgm\_id), constraint Pgm\_uk Unique

(Pgm\_name), constraint Pgm\_id\_ck check (Pgm\_id in ('CPA','CPD')), c

onstraintPg\_sem\_ck check (No\_semesters>=0))

create table Student(Campus\_no char(1), Stud\_no Numeric(3), Stud\_lna

me char(15), Stud\_fname char(15), Stud\_pgm char (3), Locker\_noNumer

ic (3), constraint Std\_pk primary key (Campus\_no, Stud\_no), constrai

ntstd\_ck check (Stud\_pgm in ('CPA')), constraint std\_uk Unique (Loc

ker\_no), constraint std\_fk foreign key (Stud\_pgm) references PGM(Pgm

\_id))

Insert the following data into your tables, be sure to enter the rows of the parent tablebefore the rows of the child table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Student | | | | | |
| 1 | 123 | Jane | Wong | CPD | 900 |
| 1 | 456 | David | Kasim | CPA | 905 |
|  | | | | | |
|  | | | | | |
| Pgm | | | | | |
| CPD | | Computer Programming | | | 4 |
| CPA | | Computer Programming and Analysis | | |  |
|  | | | | | |

If you had trouble inserting the last row, read on! When we want the system to insert a default value into a field We must use the INSERT command and specify exactly which fields we wish to supply values for. Enter the command as follows:  
  
INSERT INTO PRMIERxxx.PGM (Pgm\_id, Pgm\_name)   
 VALUES(‘CPA’, ‘Computer Programming and Analysis’)  
  
Now run a select command on the PGM file. What is listed as the No of Semesters for the CPA program?

\_\_\_\_\_\_\_\_\_

What happens if you enter the following data into the Student table?

(1, 321, ‘Joe’ , ‘Chen’, ‘CPC’, 906)  
 (1, 654, ‘Cindy’, ‘Wong’, ‘CPA’, 905)

You should NOT be able to insert these two records!The Foreign Key should prevent the Joe Chen row from being added because the parent record is missing on the PGM table. The Cindy Wong row should be rejected because of a UNIQUE violation on Locker\_No.If the rows were not rejected then delete the two rows that you have just entered which should have been rejected using DELETE FROM STUDENT WHERE … on the green screen. You could also have deleted them using the Navigator. Correct your FOREIGN KEY and UNIQUE constraints and then try to enter the two records again.

**4. Granting Permissions giving other users access to your database and tables**

To give permission to use your schema objects to another user, **you must give permission for both the collection AND the tables and/or VIEWs.**

There is no SQL statement to allow access to your schema (collection, Library)

You can do this in System i Navigator or in Client Access at the command line with a native command.

We will do this in Client Access.

Instructor Id’s are as follows:

Tanvir Alam ATANVIR

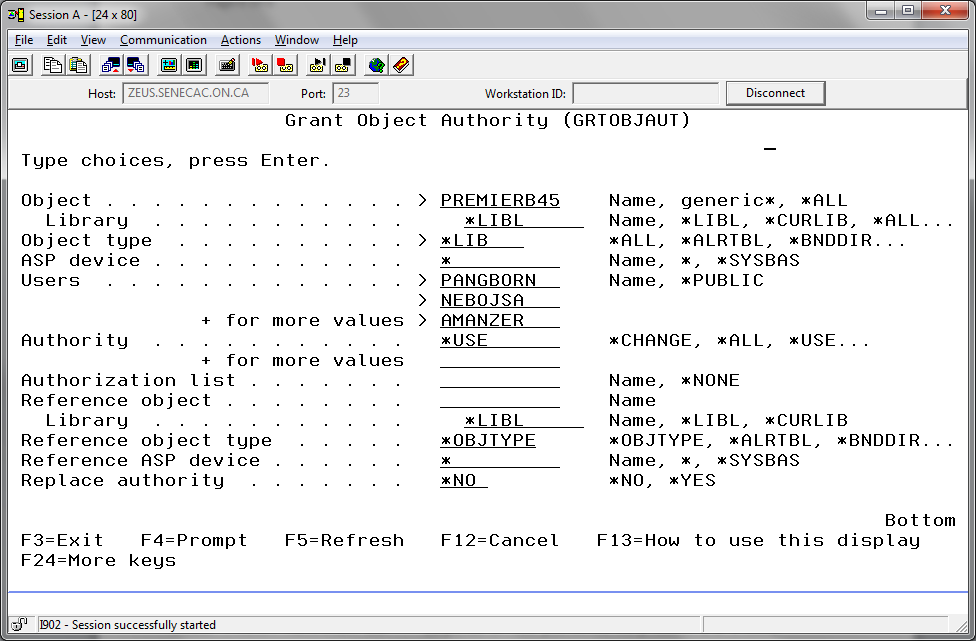
Lydia Li Lydia\_Li

Nebojsa Conkic NEBOJSA  
Russell Pangborn PANGBORN

GRTOBJAUT (F4)

The screenshot shows minimal access being granted to the library (schema). That means instructors can only access the objects in your collection with whatever specific authorities you have given to the object.

So if you have granted SELECT to the customer table in PRMIER???, they need at least \*USE access to the collection to be able to see the rows in the customer table.



GRTOBJAUT OBJ(PRMIERB45) OBJTYPE(\*LIB) USER(ATANVIR PANGBORN NEBOJSA) AUT

(\*USE)

Let’s make sure everyone else does not have access to your collection

RVKOBJAUT OBJ(PRMIERB45) OBJTYPE(\*LIB) USER(\*PUBLIC) AUT(\*ALL)

We want to be able to use SQL for the various types of authority allowed to a selection of usersfor access to the Customer and SalesRep tables in your collection. (these ids that have not been handed out to students, **if the ids do not exist, then work on DA201B45. The purpose is to learn how to grant and revoke permissions. So, if the ids starting with BA do not exist, then the permission display from the GUI i.e., the next 4 images would not match exactly. However you learn the process.)**

Remember - Whatever access you give to these ids for the tables in the collection, it is not really available until you grant access to the collection itself for that id.

Try these security setting activities in interactive SQL:

SET SCHEMA PRMIER???

GRANT SELECT ON CUSTOMER TO BA201B40, BA201B41

GRANT of authority to CUSTOMER in PRMIERB45 completed

GRANT INSERT, SELECT, UPDATE ON CUSTOMER TO BA201B42, BA201B43

GRANT DELETE, INDEX ON CUSTOMER TO BA201B44, BA201B45

GRANT ALTER, REFERENCES ON CUSTOMER TO BA201A40

GRANT REFERENCES(SALES\_REP\_NUMBER) ON SALESREP TO BA201A45

GRANT ALTER, REFERENCES ON CUSTOMER TO BA201A45

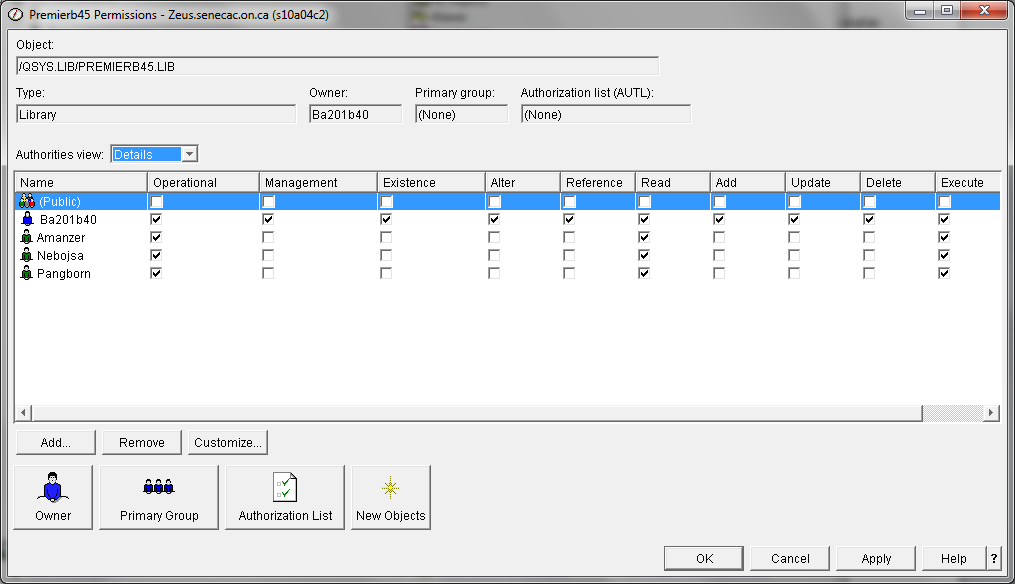
GRANT UPDATE(LAST\_NAME, STREET, CITY, STATE, ZIP\_CODE) ON CUSTOMER TO BA201A44

You can look at these settings easily with DSPOBJAUT or with System i Navigator.

Let’s try viewing these settings we did with native commands and SQLusing the GUI.

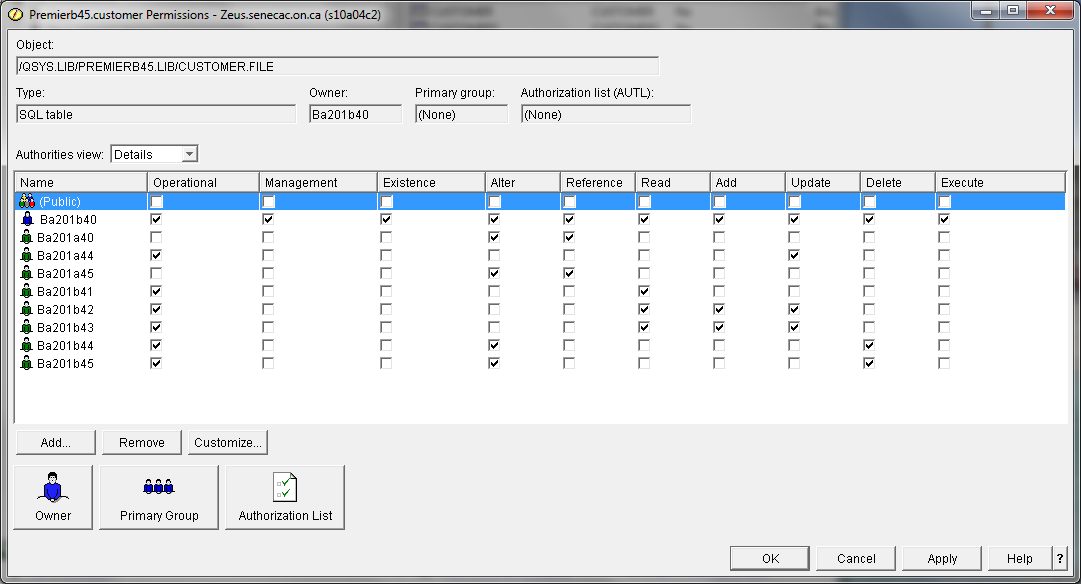
In System I Navigator, open up the Database Schemas and **right click**on your PRMIER??? Collection.

Select Permissions. **(if your collection does not show up, you need to Select Schemas to Display first)**

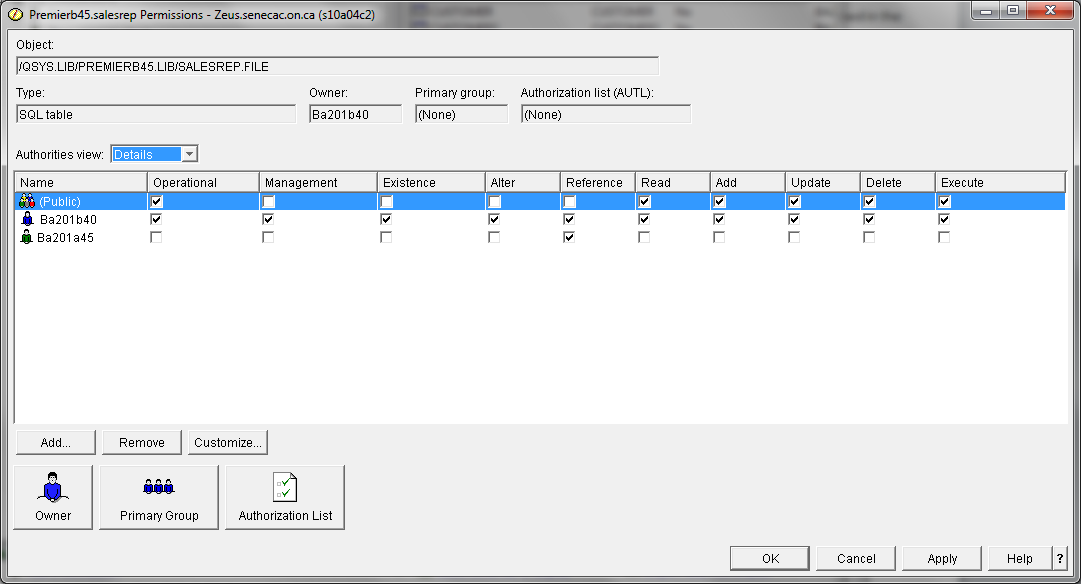


Now, open up your schema, select Tables and in the right pane right click on the Customer Table to select permissions for that specific object.

Customer should look like this:



SalesRep should look like this:



Close up your iSeries Navigator snapshot for authorities.

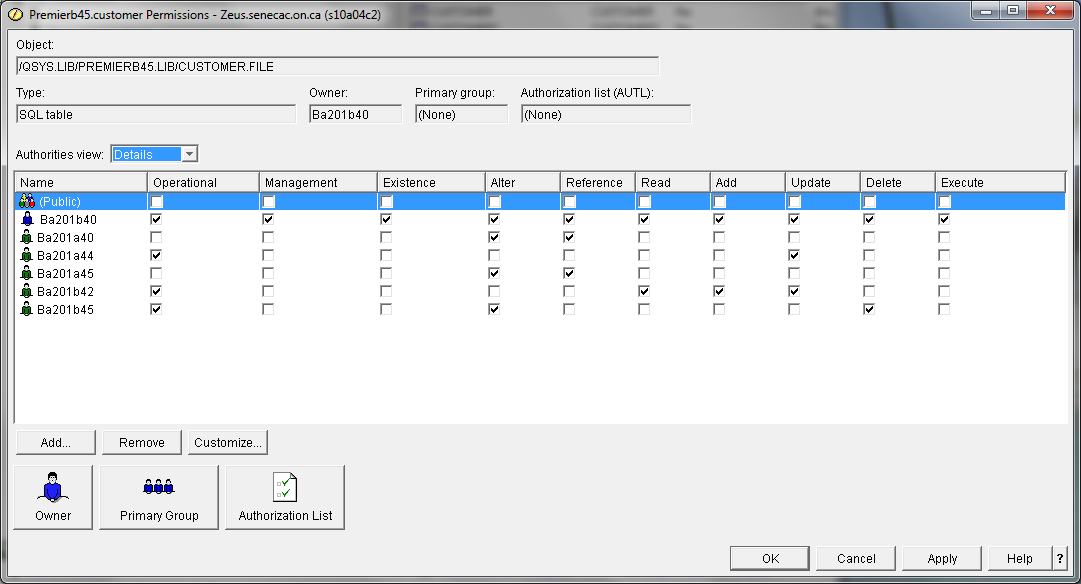
Let’s remove some authorities in interactive SQL

REVOKE SELECT ON CUSTOMER FROM BA201B41

REVOKE INSERT,SELECT, UPDATE ON CUSTOMER FROM BA201B43

REVOKE DELETE,INDEX ON CUSTOMER FROM BA201B44

Reopen your view of permissions for CUSTOMER in PRMIER????



This final screenshot will be handed in.

To Hand-in:

1. SQL statement that produces the new order info report including the Part Description

2. A screenshot from the DSPFD command to show the constraints on the STUDENT and PGM tables (refer to lab4 for DSPFD command).

3. A screen shot showing the security settings for the CUSTOMER table in PRMIER???.

4. A screen shot showing that the instructors have access to your collection, but no other ids have access thus voiding the access that shows for the customer table screenshot for the selection of student accounts specified.

**REFERENCE PAGE:**

**The following privileges can be used with the SQL GRANT statement for a table**:

ALL

Grants ALL table (file) privileges you currently have for all the tables or views (logical files) named in the ON clause. If ALL is not selected, you must use one or more of the keywords that follow.

Note: An error will result if the ALL privilege is selected with any of the other privileges.

ALTER

Grants the privilege to use the ALTER TABLE, COMMENT ON, and LABEL ON statements on the tables (files) or views (logical files) names in the ON clause.

DELETE

Grants the privilege to use the DELETE statement on the table (physical file) or view (logical file) named in the ON clause.

INDEX

Grants the privilege to use the CREATE INDEX statement on the table (physical file) named in the ON clause. Views may not be specified when granting INDEX privileges.

INSERT

Grants the privilege to use the INSERT statement on the table (physical file) or view (logical file) named in the ON clause. Shows up as ADD in iSeries Navigator and with the native DSPOBJAUT command.

REFERENCES

The user specified is granted REFERENCES privileges to the tables or views (logical files) specified in the ON clause.

REFERENCES (column)

The user specified is granted REFERENCES privileges to the columns selected from the tables specified in the ON clause.

SELECT

Grants the privilege to use the SELECT statement on the table (physical file) or view (logical file) named in the ON clause. Shows up as READ in iSeries Navigator and with the native DSPOBJAUT command.

UPDATE

Grants the privilege to use the UPDATE statement on the table (physical file) or view (logical file) named in the ON clause.

UPDATE (column)

Grants the privilege to use the UPDATE statement on columns selected from the table (physical file) named in the ON clause.

WITH GRANT OPTION

The following privileges can be selected:

Y=Yes

Allows the specified users to grant privileges on the specified tables (files) to other users. Grants the user \*OBJMGT authority to the table (file).

N=No

Does not give users the right to grant privileges on the specified tables (files). Users cannot grant privileges unless

they have received that authority from some other source.