

# **DATABASE SYSTEMS: GUIDELINES FOR THE COMPUTER LABS**

## **Contents**

Background Information .....	2
Lab Scenario.....	3
Lab 1. Configuring Oracle SQL Developer .....	4
Activity 1. Create Oracle SQL Developer Shortcut.....	4
Activity 2. Configure Oracle SQL Developer .....	4
Activity 3. First Run of Oracle SQL Developer .....	5
Lab 2. Creating Tables .....	7
Activity 4. Create Tables.....	7
Activity 5. Define Primary Keys.....	8
Activity 6. Define Foreign Keys .....	10
Activity 7. Define NOT NULL and DEFAULT Integrity Constraints.....	10
Activity 8. Define CHECK Integrity Constraints .....	11
Activity 9. Define UNIQUE Integrity Constraints .....	12
Lab 3. Modifying Tables.....	14
Activity 10. Insert Rows into Table .....	14
Activity 11. Update Rows in Table.....	16
Activity 12. Delete Rows from Table .....	16
Lab 4. Querying Tables.....	17
Activity 13. Construct Simple Select Query .....	17
Activity 14. Construct Simple Select Query without Duplication of Rows .....	18
Activity 15. Construct Simple Select Query with Aggregation Functions.....	19
Activity 16. Construct Simple Select Query with Grouping .....	20
Activity 17. Construct Complex Query .....	21
Lab 5. Triggers .....	23
Activity 18. Create and manage triggers.....	23
Lab 6. Privileges and roles .....	25
Activity 19. Granting and revoking privileges to users .....	25
Activity 20. Managing roles.....	26
Introduction	

## Background Information

All the computer labs are held by means of **Personal Virtual Computer (PVC)** system (i.e. you should log in PVC before starting a lab). PVC installation instructions are available at PVC homepage: <https://pvc.susu.ac.ru/>.

**Computer labs** are aimed to learning object-relational features of Oracle Database Management System (DBMS). You will use the following free software that has been installed within PVC:

- Oracle XE (eXpress Edition) DBMS as a database server and
- Oracle SQL Developer as a client program.

Doing a lab, **ask instructor to help** in case of any technical problem. Having done a lab, **ask instructor to verify results** of your lab.

**Useful links** concerning software (you may use these URLs to see or download content for self-study):

- Oracle Database in Wikipedia:  
[http://en.wikipedia.org/wiki/Oracle\\_Database](http://en.wikipedia.org/wiki/Oracle_Database;);
- Oracle SQL Developer in Wikipedia:  
[http://en.wikipedia.org/wiki/Oracle\\_SQL\\_Developer](http://en.wikipedia.org/wiki/Oracle_SQL_Developer);
- Oracle XE Documentation:  
<http://www.oracle.com/technetwork/products/express-edition/documentation/>;
- Oracle SQL Developer Documentation:  
[http://docs.oracle.com/cd/E12151\\_01/doc.150/e12152.pdf](http://docs.oracle.com/cd/E12151_01/doc.150/e12152.pdf);
- Download Oracle XE:  
<http://www.oracle.com/technetwork/products/express-edition/downloads/>;
- Download SQL Developer:  
<http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/>.

## Lab Scenario

You are to implement a database application for some model domain. There are Suppliers, Parts and Supplies in this domain.

**Supplier** has the following attributes:

- SID – identity, character string of no more than 5 symbols (e.g. 'S0001', 'S0002', etc.);
- Name – name of supplier, character string of no more than 20 symbols;
- City – city of supplier, character string of no more than 20 symbols;
- Rating – non-negative integer indicator of supplier's reliability with the following semantic:
  - 0..10 – Problematic
  - 11..15 – Neutral
  - 16..20 – Acceptable
  - 21..25 – Reliable.

**Part** has the following attributes:

- PID – identity, character string of no more than 5 symbols (e.g. 'P0001', 'P0002', etc.);
- Name – name of part, character string of no more than 20 symbols;
- City – city of part, character string of no more than 20 symbols;
- Price – in EUR, positive real;
- Weight – in grams, positive real;
- Color – character string, one of the following: (Red, Green, Blue, Grey, Brown, White, Black).

**Supply** has the following attributes:

- SID – identity of a respective supplier;
- PID – identity of a respective part;
- Qty – positive integer, number of parts in the supply.


## Lab 1. Configuring Oracle SQL Developer

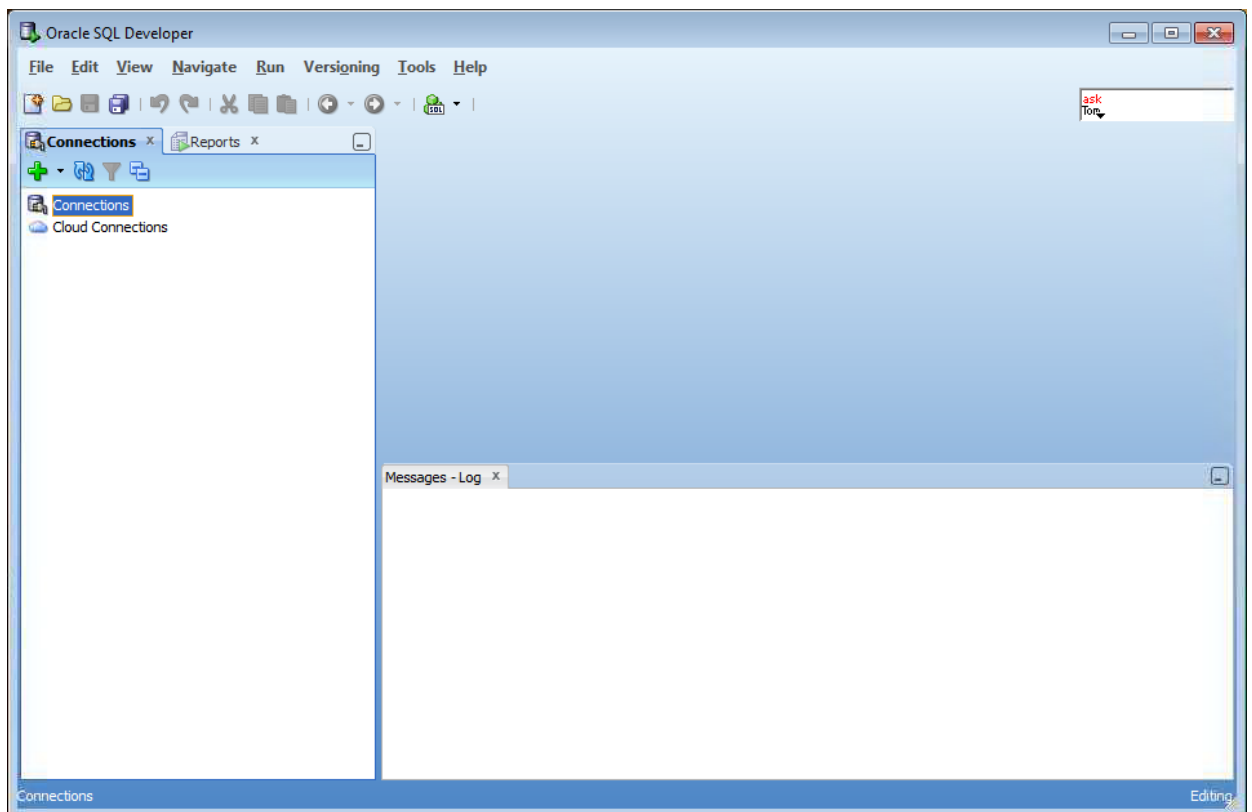
**OBJECTIVE.** In this lab you will learn how to configure and run Oracle SQL Developer.

### Activity 1. Create Oracle SQL Developer Shortcut

1. With Windows Explorer, open the folder C:\DataBases\sqldeveloper\.
2. Grab sqldeveloper.exe file with right mouse button, then drag and drop it to your desktop. Rename shortcut to 'SQL Developer'.

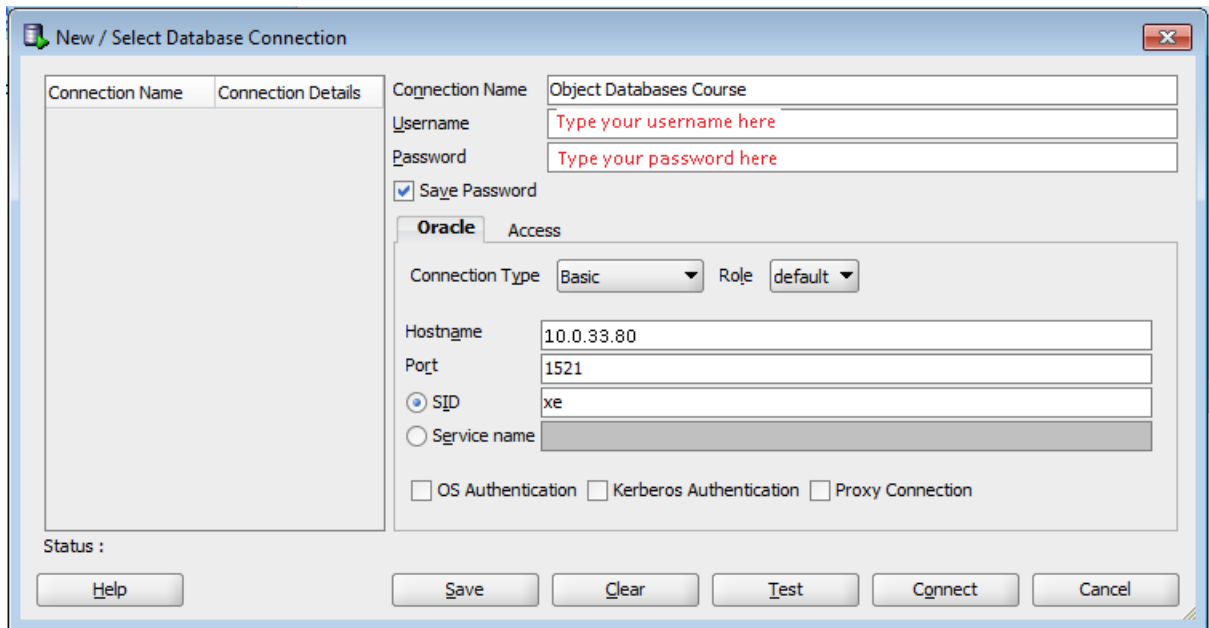
### Activity 2. Configure Oracle SQL Developer

1. Run Oracle SQL Developer using shortcut.
2. At the Oracle SQL Developer window (see Fig. 1) go to the Connections tab and click the  button.



**Fig. 1. Oracle SQL Developer window**

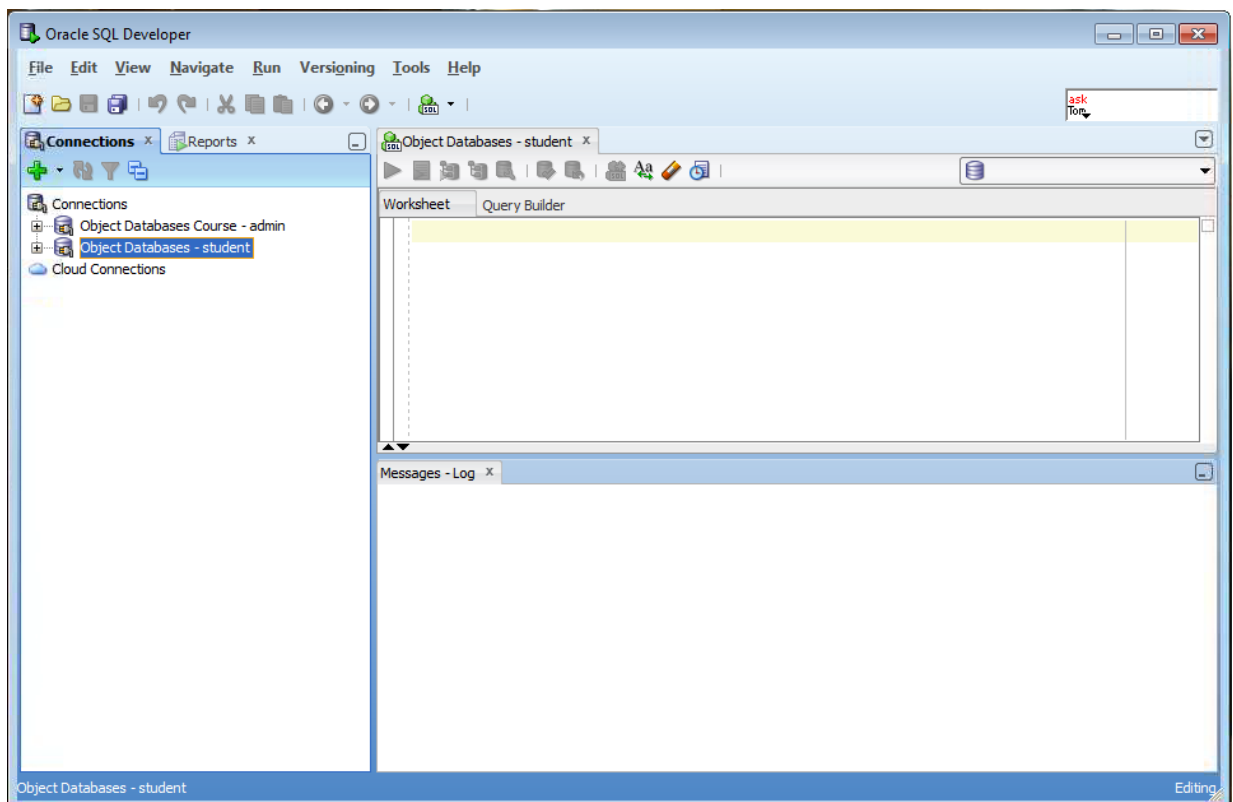
3. At the New / Select Database Connection dialog fill the form as show at the Fig. 2 (for your username and password ask Instructor). Click Save button. Click Test button, then check that Status (left bottom corner, under the Help button) is Success.



**Fig. 2. New / Select Database Connection window**

### **Activity 3. First Run of Oracle SQL Developer**

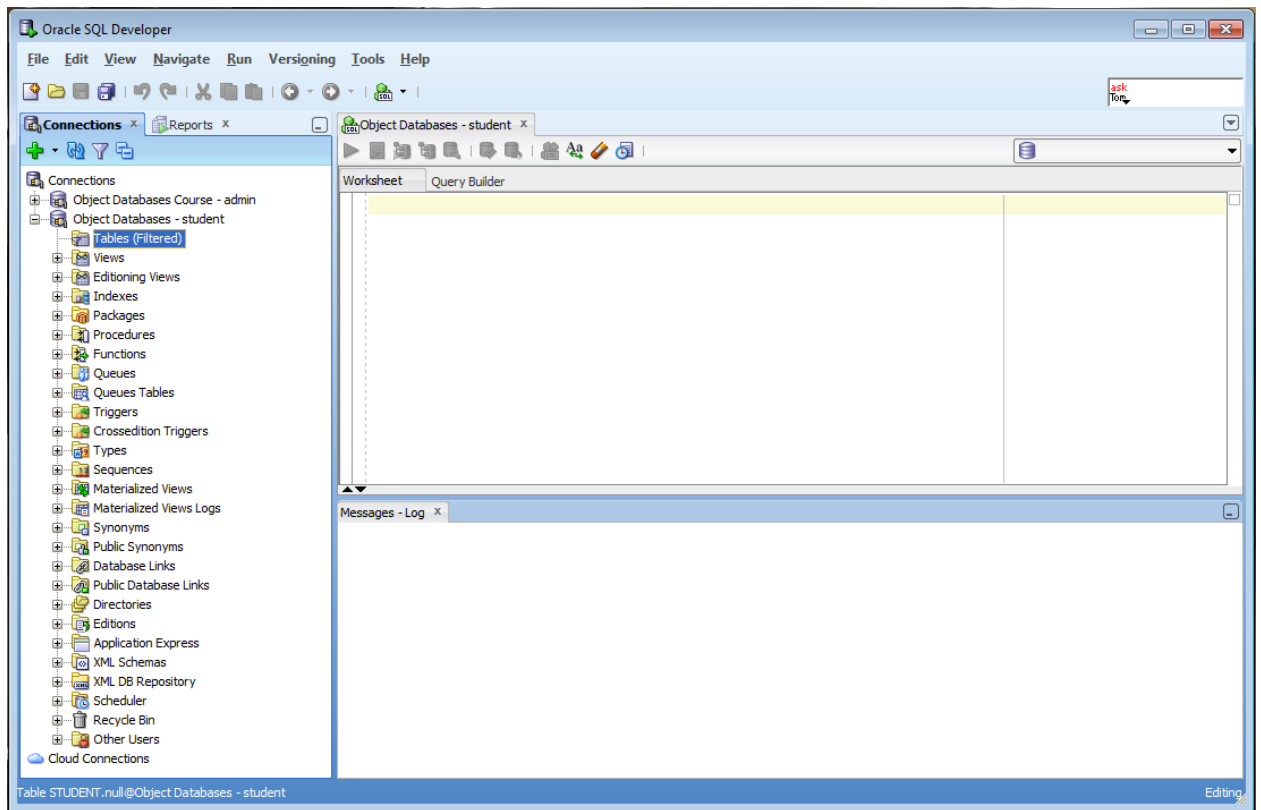
1. Click Connect button in New / Select Database Connection dialog, check that you have successfully connected to Oracle Server and Oracle SQL Developer looks like at Fig. 3.



**Fig. 3. Oracle SQL Developer after establishing a connection**

2. Click the + sign in the Connection tree to drill down your connection contents, check that Oracle SQL Developer looks like at Fig. 4. Click some + signs in the

tree of your connection (e.g. tables, views, etc.) to verify that they are empty (no objects have been created yet).



**Fig. 4. Connection contents at first run**

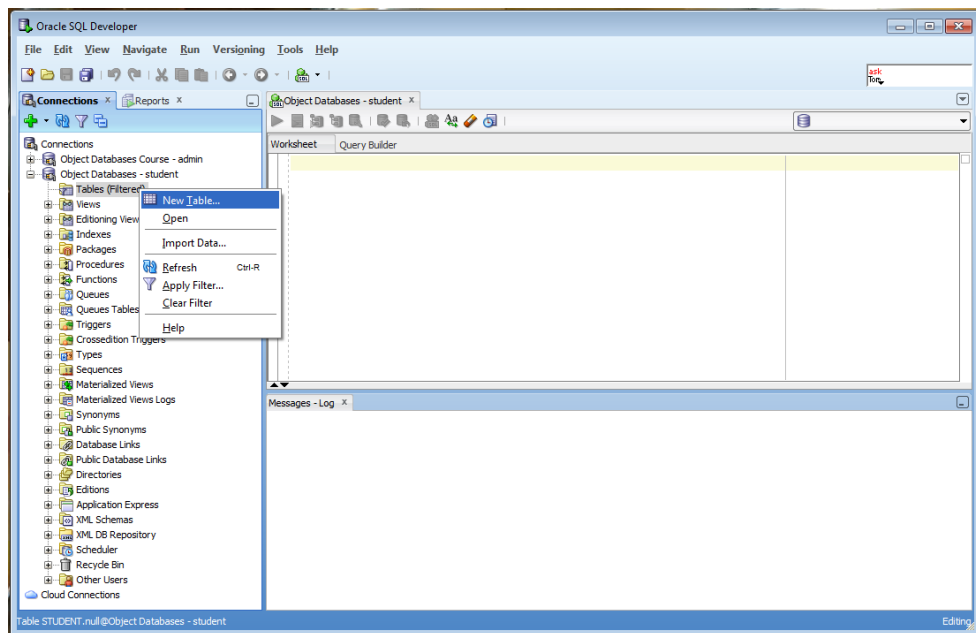
**ASK INSTRUCTOR** to verify the results of your lab.

## Lab 2. Creating Tables

**OBJECTIVE.** In this lab you will learn how to create table and define various integrity constraints of its columns in Oracle SQL Developer.

### Activity 4. Create Tables

1. Click right mouse button on the Tables branch of connection tree (see Fig. 3) and choose the New Table command.



**Fig. 5. Creating a table using Oracle SQL Developer wizard**

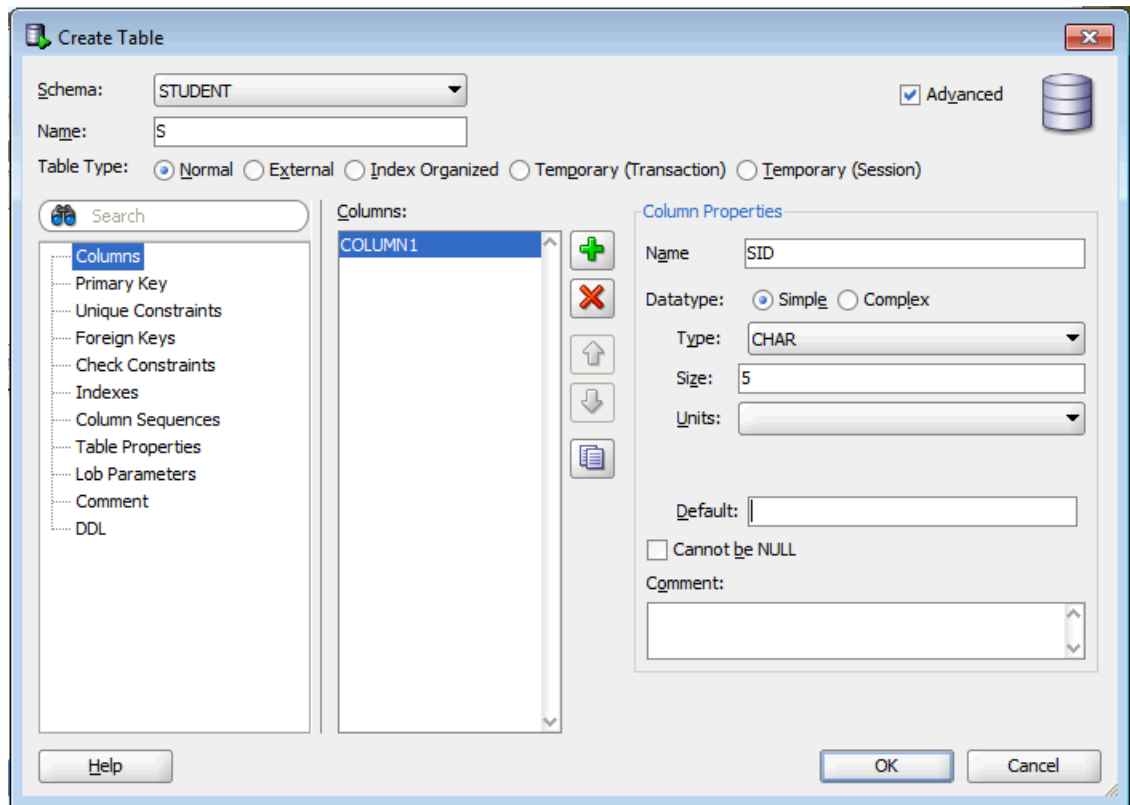
2. Fill the Create table form as shown at the Fig. 6.

A screenshot of the 'Create Table' dialog box in Oracle SQL Developer. The 'Schema' dropdown is set to 'STUDENT'. The 'Name' text field contains 'S'. The 'Advanced' checkbox is unchecked. The 'Table' tab is selected, showing a table with one column: 'COLUMN1' of type 'VARCHAR2' with a size of '20'. The 'Not Null' and 'Primary Key' checkboxes are both unchecked. At the bottom, there are buttons for 'Add Column', 'Remove Column', 'Help', 'OK', and 'Cancel'.

Column Name	Type	Size	Not Null	Primary Key
COLUMN1	VARCHAR2	20	<input type="checkbox"/>	<input type="checkbox"/>

**Fig. 6. Create table form**

3. Check the Advanced box and add the SID column as shown at Fig. 7.



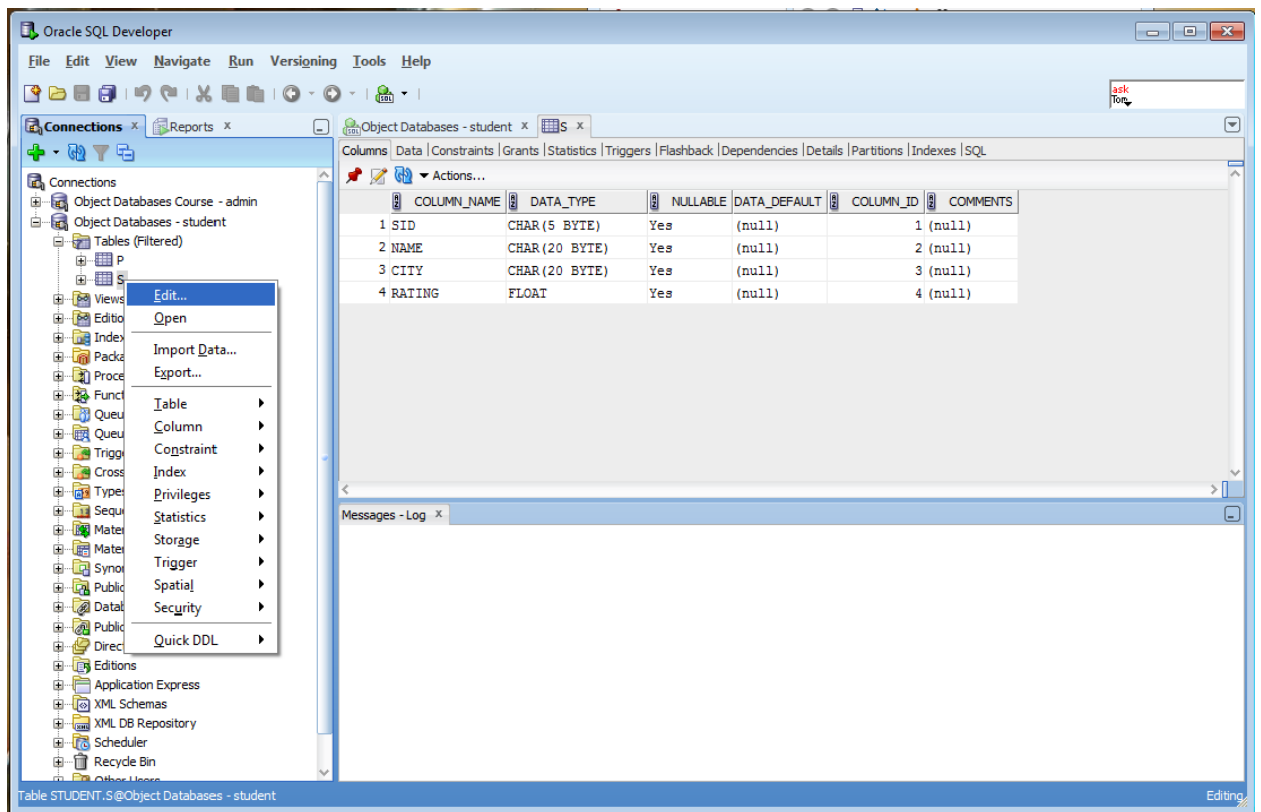
**Fig. 7. Add column form**

4. Proceeding similarly, add to the S table the following columns:
- Name char(20)
  - City char(20)
  - Rating int.
5. Proceeding similarly, create the P table with the following columns:
- PID char(5)
  - Name char(20)
  - City char(20)
  - Price real
  - Weight real
  - Color char(10).
6. Proceeding similarly, create the SP table with the following columns:
- PID char(5)
  - SID char(5)
  - Qty int.

### **Activity 5. Define Primary Keys**

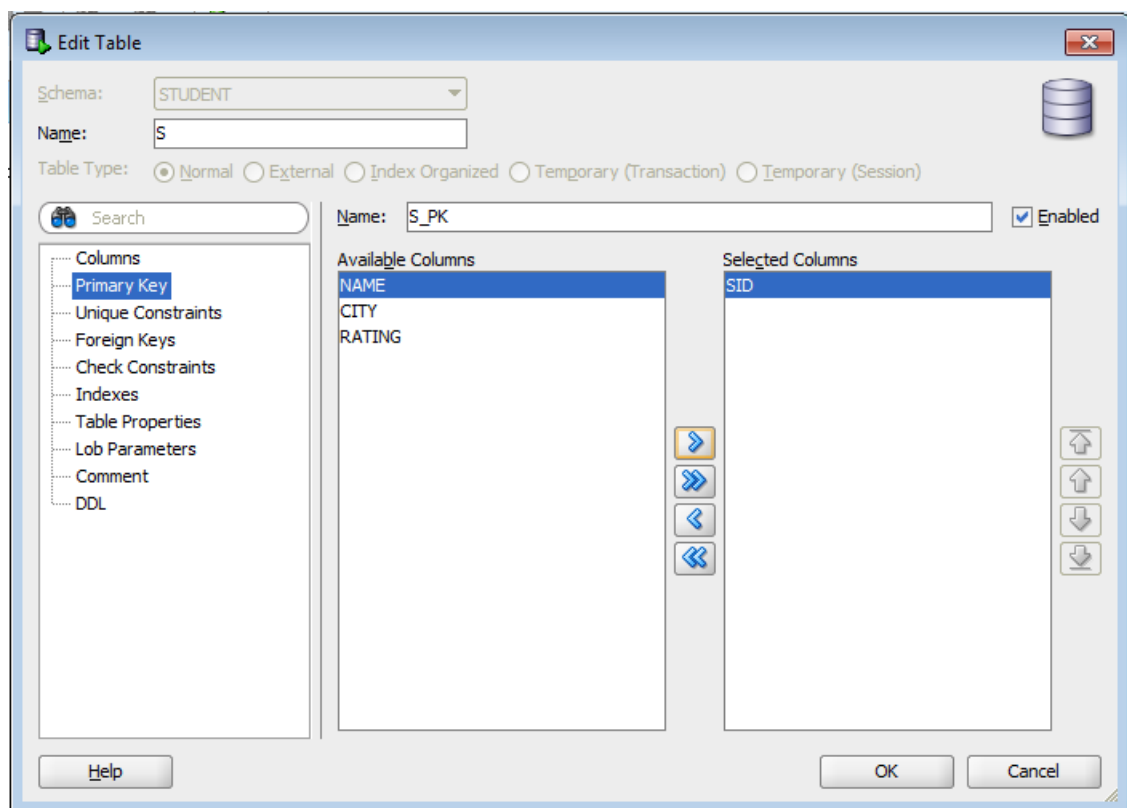
1. Click right mouse button on the S table of Tables branch of connection tree (see Fig. 8) and choose the Edit command.





**Fig. 8. Edit table structure**

2. In the Columns pane of the Edit Table form click the Primary Key button, then in the Available columns pane select the SID column and click the > button as shown at the Fig. 9.



**Fig. 9. Defining primary key**

3. Proceeding similarly, define the PID column as primary key of the P table.
4. Proceeding similarly, define *both* SID and PID columns as *complex* primary key of the SP table.

### Activity 6. Define Foreign Keys

1. Proceeding similarly as above, call the Edit Table form for the SP table. In the Columns pane of the Edit Table form click the Foreign Key branch and fill the form as shown at the Fig. 10.

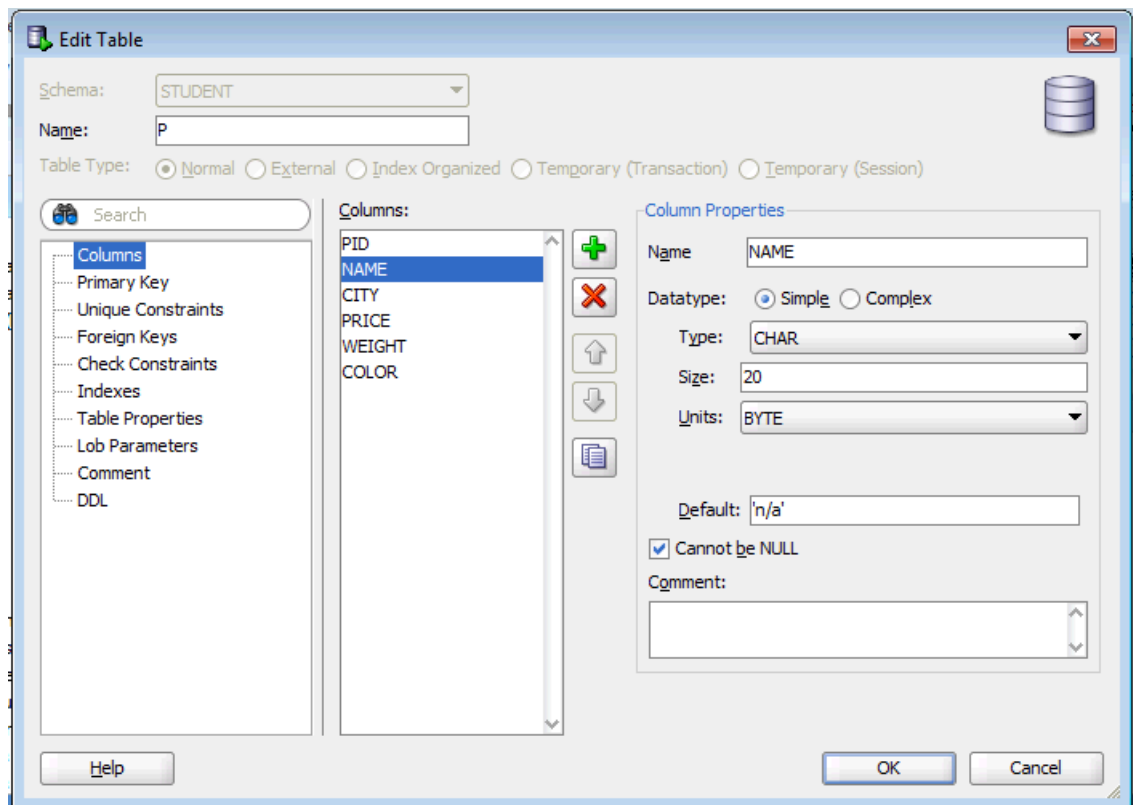
The screenshot shows the 'Edit Table' dialog box for the 'SP' table in the 'STUDENT' schema. The 'Foreign Keys' tab is selected in the left pane. The 'Foreign Keys' list shows 'SP\_S\_FK'. The 'Name' field is 'SP\_S\_FK', 'Referenced Schema' is 'STUDENT', 'Referenced Table' is 'S', and 'Referenced Constraint' is 'S\_PK'. The 'Associations' table shows 'SID' as both the local and referenced column. The 'On Delete' action is set to 'CASCADE'.

Fig. 10. Defining foreign key

2. Proceeding similarly, define the PID column as foreign key referencing the PID column of the P table with cascading deletion; name this constraint as SP\_P\_FK.

### Activity 7. Define NOT NULL and DEFAULT Integrity Constraints

1. Proceeding similarly as above, call the Edit Table Columns form for the P table and fill the form for the Name column as shown at the Fig. 11.
2. Proceeding similarly, define the following integrity constraints for the P table columns:
  - Name: NOT NULL, 'n/a' by default
  - City: NOT NULL, 'n/a' by default
  - Color: NOT NULL, 'n/a' by default
  - Price: NOT NULL
  - Weight: NOT NULL.

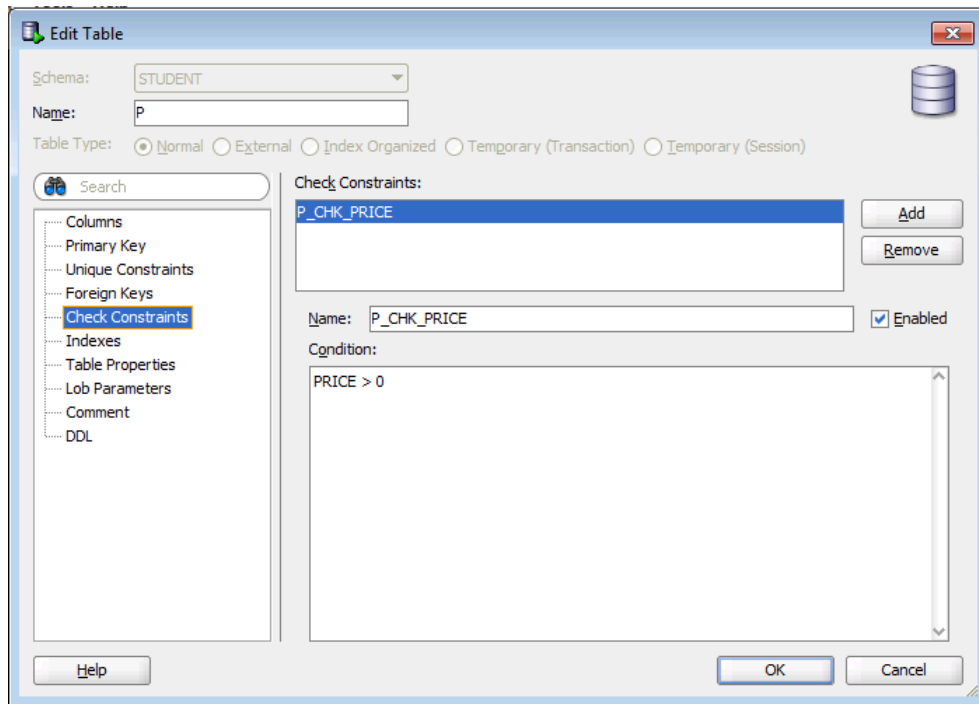


**Fig. 11. Defining NOT NULL and DEFAULT constraints**

3. Proceeding similarly, define the following integrity constraints for the S table columns:
  - City: NOT NULL, 'n/a' by default
  - Rating: NOT NULL.
4. Proceeding similarly, define the following integrity constraint for the SP table column:
  - Qty: NOT NULL, 1 by default.

### **Activity 8. Define CHECK Integrity Constraints**

1. Proceeding similarly as above, call the Edit Table Columns form for the P table. In the Columns pane of the Edit Table form click the Check Constraints branch, then click Add button and fill the form as shown at the Fig. 12.
2. Proceeding similarly, define the following integrity constraints for the P table columns:
  - Weight > 0, name this constraint as P\_CHK\_WEIGHT
  - Color in ('n/a', 'Red', 'Green', 'Blue', 'Black', 'White', 'Grey', 'Brown'), name this constraint as P\_CHK\_COLOR.
3. Proceeding similarly, define the following integrity constraint for the SP table column:
  - Qty > 0, name this constraint as SP\_CHK\_QTY

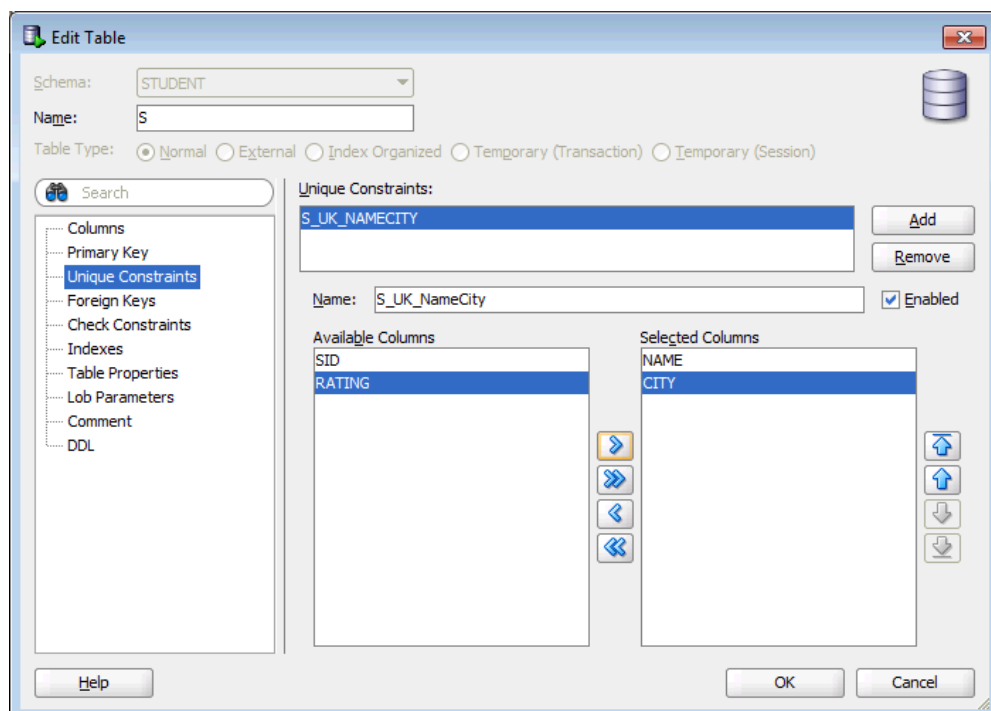


**Fig. 12. Defining CHECK constraints**

4. Proceeding similarly, define the following integrity constraint for the S table column:
  - Rating  $\geq 0$  and Rating  $\leq 25$ , name this constraint as S\_CHK\_RATING.

### Activity 9. Define UNIQUE Integrity Constraints

1. Proceeding similarly as above, call the Edit Table Columns form for the S table. In the Columns pane of the Edit Table form click the Unique Constraints branch, then click Add button and fill the form as shown at the Fig. 12.



**Fig. 13. Defining UNIQUE constraints**

2. Proceeding similarly, define the following integrity constraint for the P table column:
  - UNIQUE (Name, City, Price, Weight, Color), name this constraint as P\_UK\_ALLCOLUMNS.

**ASK INSTRUCTOR** to verify the results of your lab.

## Lab 3. Modifying Tables

**OBJECTIVE.** In this lab you will learn how to insert, update and delete rows of a table in Oracle SQL Developer.

### Activity 10. Insert Rows into Table

1. Proceeding similarly as above, click the S table in the connection tree, then click Data tab as shown at the Fig. 14.

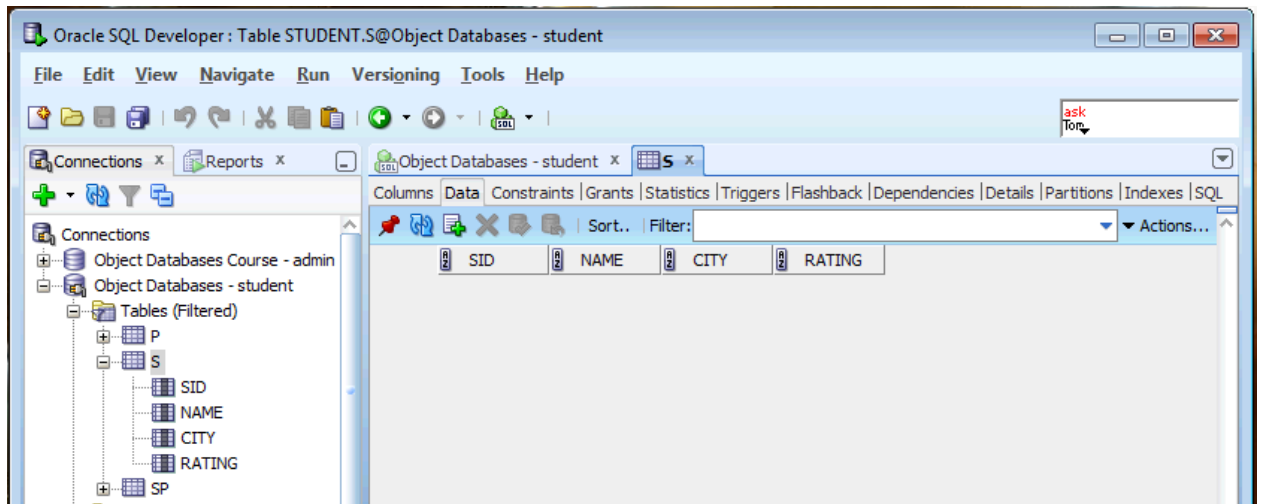




Fig. 14. Data of a table

2. Click the  (Insert Row) button, then add contents of a new row of the S table as shown at the Fig. 15. Then click  (Commit Changes) button.

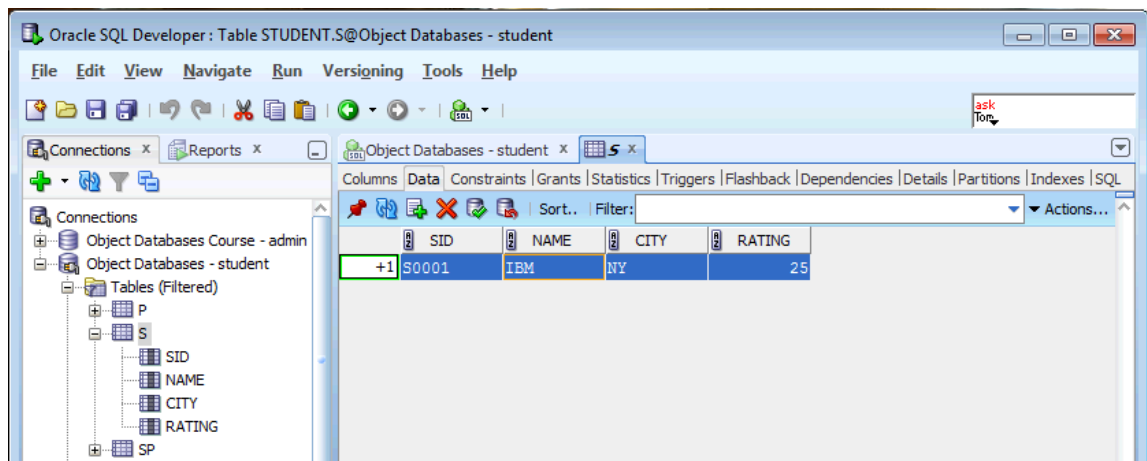


Fig. 15. Inserting a record into a table

3. Proceeding similarly, insert the following rows into the S table:

<i>SID</i>	<i>Name</i>	<i>City</i>	<i>Rating</i>
S0002	IBM	NY	20
S0003	Raznoexport	Moscow	14
S0004	Chanel	Paris	18
S0005	H&H	Chelyabinsk	15
S0006	CyberGun	Moscow	19

<i>SID</i>	<i>Name</i>	<i>City</i>	<i>Rating</i>
S0007	CocaCola	Paris	14
S0008	Marks	London	21
S0009	Zorba LTD	Athens	10
S0010	Spencer	London	22
S0201	UralCloud	Chelyabinsk	17

4. Proceeding similarly, insert the following rows into the P table:

<i>PID</i>	<i>Name</i>	<i>City</i>	<i>Price</i>	<i>Weight</i>	<i>Color</i>
P0201	Push-clip	Chelyabinsk	1.7	5	Red
P1002	CPU	NY	200	10	Grey
P1003	Bolt	Moscow	1.5	25	Grey
P1004	Clip	Paris	1.8	5	Green
P1005	Timber	Chelyabinsk	150	500	Brown
P1006	Tile	Moscow	190	1000	White
P1007	HDD	Paris	140	100	Black
P1008	Screw-nut	London	2.1	30	Blue
P1009	Screw	Athens	2.0	20	White
P1010	Angle-part	London	30	50	Black

5. Proceeding similarly, insert the following rows into the SP table:

<i>SID</i>	<i>PID</i>	<i>Qty</i>	<i>SID</i>	<i>PID</i>	<i>Qty</i>	<i>SID</i>	<i>PID</i>	<i>Qty</i>
S0003	P1003	1400	S0003	P1002	1400	S0002	P1003	1400
S0009	P1009	1000	S0009	P1005	1000	S0002	P1005	1000
S0005	P1009	1500	S0005	P1002	1500	S0005	P1005	1500
S0002	P1002	2000	S0002	P1004	2000	S0005	P1003	2000
S0004	P1004	1800	S0004	P1010	1800	S0004	P1007	1800
S0008	P1008	2100	S0008	P1007	2100	S0008	P1005	2100
S0007	P1007	1100	S0007	P1005	1100	S0201	P1007	1100
S0006	P1005	1090	S0006	P1006	1090	S0006	P1002	1090
S0010	P1010	2250	S0010	P1002	2250	S0010	P1005	2250
S0201	P0201	1700	S0201	P1010	1700	S0201	P1002	1700

6. Proceeding similarly, try to insert the following rows into the S table. Verify that it is impossible due to integrity constraint violations. For every rejected row explain what integrity constraint was violated and why (see Activities 5-9 for help).

<i>SID</i>	<i>Name</i>	<i>City</i>	<i>Rating</i>
S0003	Vegetables Inc.	Moscow	15
S0030	Vegetables Inc.	Moscow	-5
S0030	Raznoexport	Moscow	20

7. Proceeding similarly, try to insert the following rows into the P table. Verify that it is impossible due to integrity constraint violations. For every rejected row explain what integrity constraint was violated and why (see Activities 5-9 for help).

<i>PID</i>	<i>Name</i>	<i>City</i>	<i>Price</i>	<i>Weight</i>	<i>Color</i>
P1040		Moscow	15	250	Grey
P1050	Screw	Athens	2.0	20	White
P1205	Timber	London	150	500	Yellow

8. Proceeding similarly, try to insert the following rows into the SP table. Verify that it is impossible due to integrity constraint violations. For every rejected row explain what integrity constraint was violated and why (see Activities 5-9 for help).

<i>SID</i>	<i>PID</i>	<i>Qty</i>
S0003	P1003	1500
	P1009	1000
S0005		1500
S0010	P1003	0
S0000	P1003	2000
S0010	P0000	5000

### Activity 11. Update Rows in Table

1. Proceeding similarly as above, call the Data tab of the P table. Update existing rows as follows (new values are underlined; simply edit old values of the respective columns and then click Commit Changes button):

<i>PID</i>	<i>Name</i>	<i>City</i>	<i>Price</i>	<i>Weight</i>	<i>Color</i>
P1002	CPU	<u>New York</u>	200	10	Grey
P1004	<u>Tie</u>	Paris	1.8	5	Green

### Activity 12. Delete Rows from Table

1. Proceeding similarly as above, call the Data tab of the SP table. Delete the following rows (simply point to the respective row and click the × button, then click Commit Changes button):

<i>SID</i>	<i>PID</i>	<i>Qty</i>	<i>SID</i>	<i>PID</i>	<i>Qty</i>	<i>SID</i>	<i>PID</i>	<i>Qty</i>
S0003	P1003	1400	S0003	P1002	1400	S0002	P1003	1400

2. Proceeding similarly, delete the following row from S table:


<i>SID</i>	<i>Name</i>	<i>City</i>	<i>Rating</i>
S0201	UralCloud	Chelyabinsk	17

Verify that the respective supplies (rows in the SP table) are deleted as well. Explain why is that.

**ASK INSTRUCTOR** to verify the results of your lab.



## Lab 4. Querying Tables

**OBJECTIVE.** In this lab you will learn how to select data from tables in Oracle SQL Developer. Please *save every query you have made in text file* using Save  button (e.g. Activity13.sql, Activity14.sql, etc.) to show it to instructor as results of this lab.

### Activity 13. Construct Simple Select Query

*The aim of this activity is to create a simple query with some selection criteria and ordering resulting rows.*

1. You are to write a query that gets a list of parts' names prices and colors where part's price more than 10 and resulting list should be ordered by name (in ascending order) and price (in descending order).

Verify that Query Builder and SQL Worksheet tabs of Oracle SQL Developer are opened as shown at the Fig. 16 (if not, use SQL button).

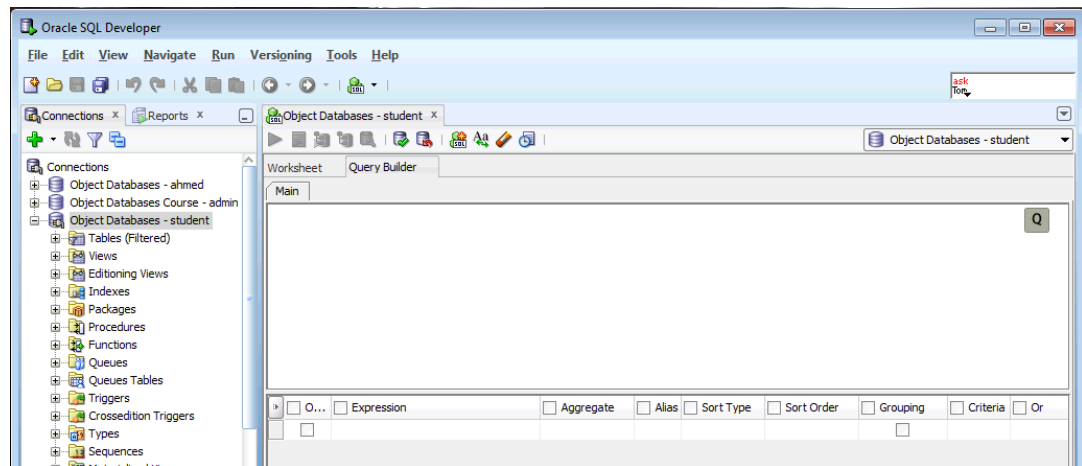


Fig. 16. SQL query constructor

2. Open the contents of the P table in the Connections tab, using left mouse button drag the P table to the Query Builder tab and drop it there. Check the results as shown at the Fig. 17.

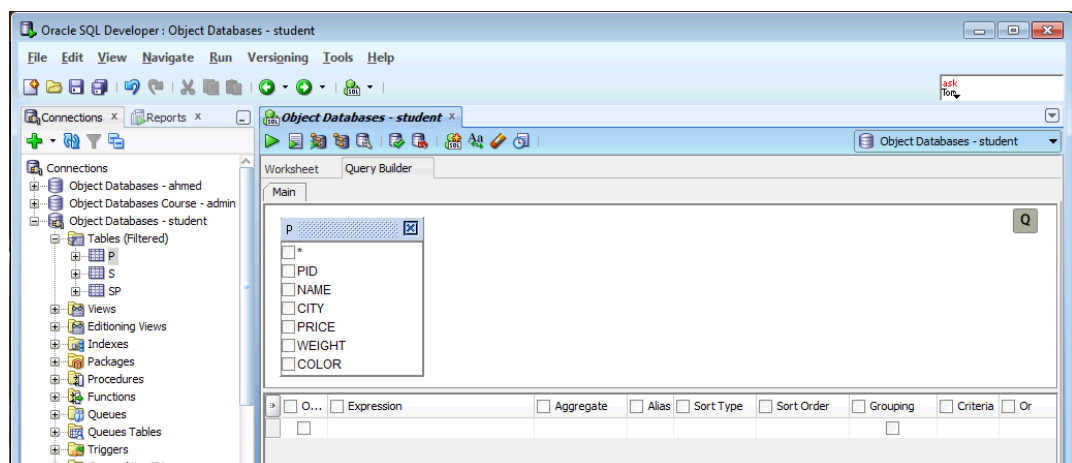
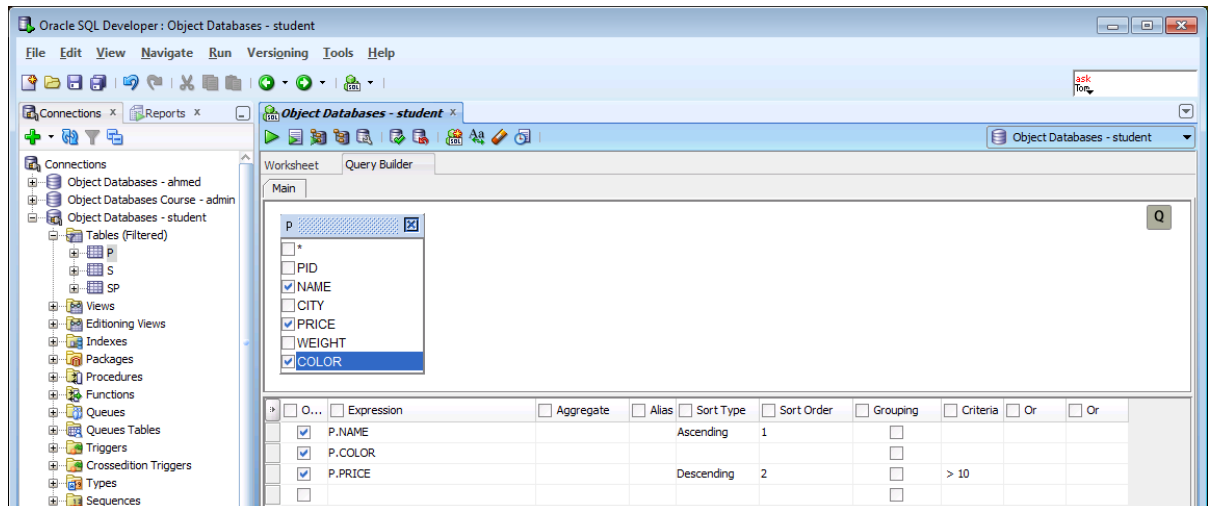


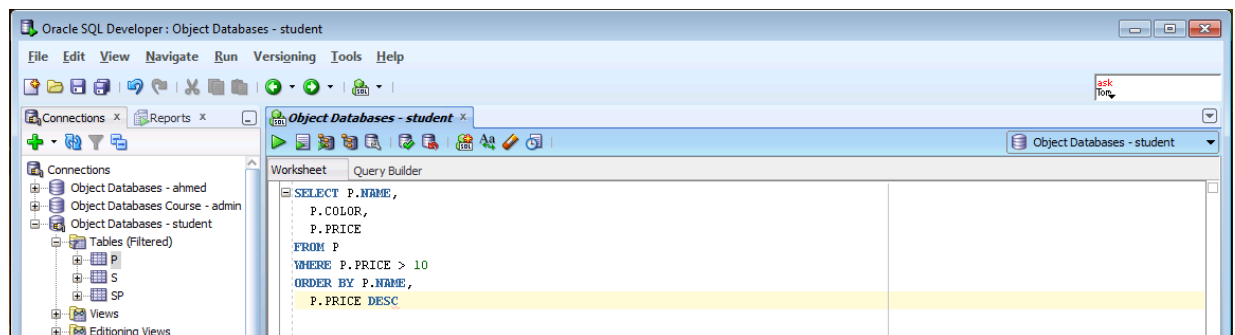
Fig. 17. Query Builder of the SQL query constructor

3. Fill the Query Builder form as shown at the Fig. 18.



**Fig. 18. Construction of the query using SQL query constructor**

4. Open the Worksheet tab and check the results as shown at the Fig. 19.



**Fig. 19. Construction of the query using SQL query constructor (pure SQL code)**

5. Click ► Run Statement button and check the results of the query in the Query results tab below.

**ASK INSTRUCTOR** to verify the results of your activity.

#### **Activity 14. Construct Simple Select Query without Duplication of Rows**

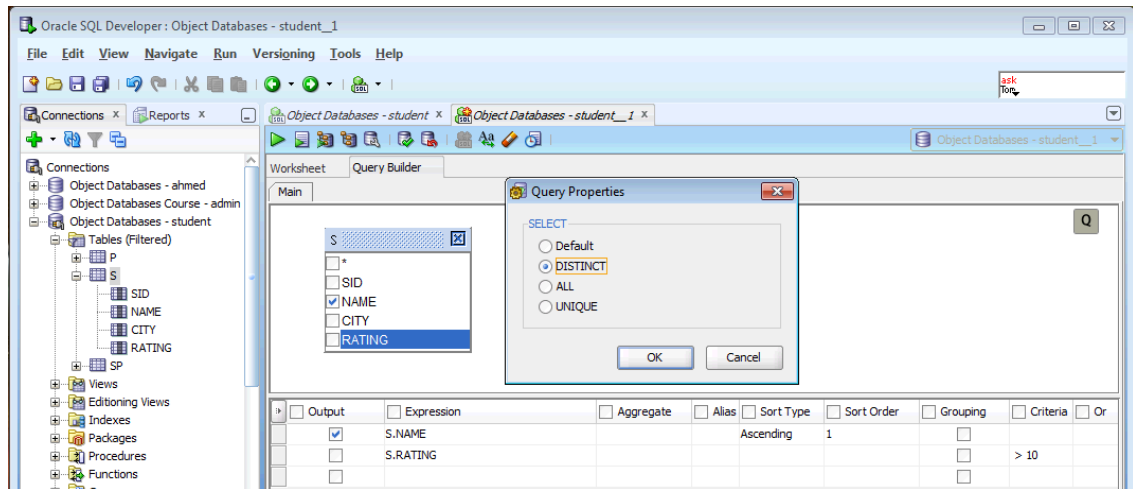
*The aim of this activity is to create a simple query without duplication of resulting rows.*

1. You are to write a query that gets a list of suppliers' cities with rating more than 10 without duplications and resulting list should be ordered by name (in ascending order).

Proceeding similarly as above create a query. Note that you should not output supplier's rating but should specify a selection criterion depending on this column.

2. Proceeding similarly as above run the query. Verify that resulting list contains duplicates.

- Click the right mouse button on the Query builder sheet and click Properties command in appeared context menu. Choose Distinct in the Query properties dialogue as shown at the Fig. 20. Click OK to save query properties.



**Fig. 20. Changing query properties**

- Switch to Worksheet to verify that pure SQL code has the DISTINCT keyword.
- Proceeding similarly as above run the query. Verify that resulting list does not contain duplicates.

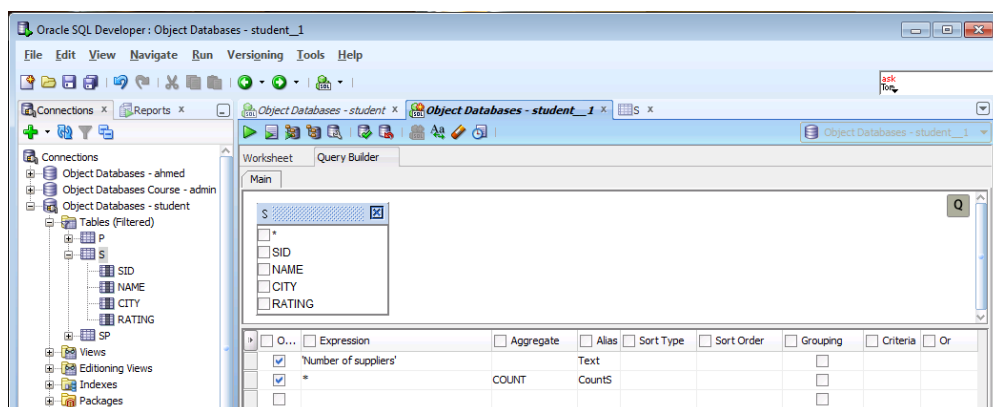
**ASK INSTRUCTOR** to verify the results of your activity.

### **Activity 15. Construct Simple Select Query with Aggregation Functions**

*The aim of this activity* is to create a simple query using aggregation functions (e.g. COUNT, SUM, MIN, MAX, etc.).

- You are to write a query that gets a number of suppliers as a table with one row and two columns: the first column should be named Text and contain a string 'Number of suppliers' and the second column should be named CountS and contain a number.

Proceeding similarly as above create a query with parameters as shown at the Fig. 21.



**Fig. 21. Query with COUNT aggregation function**

- Switch to Worksheet to verify that pure SQL code has the COUNT keyword.

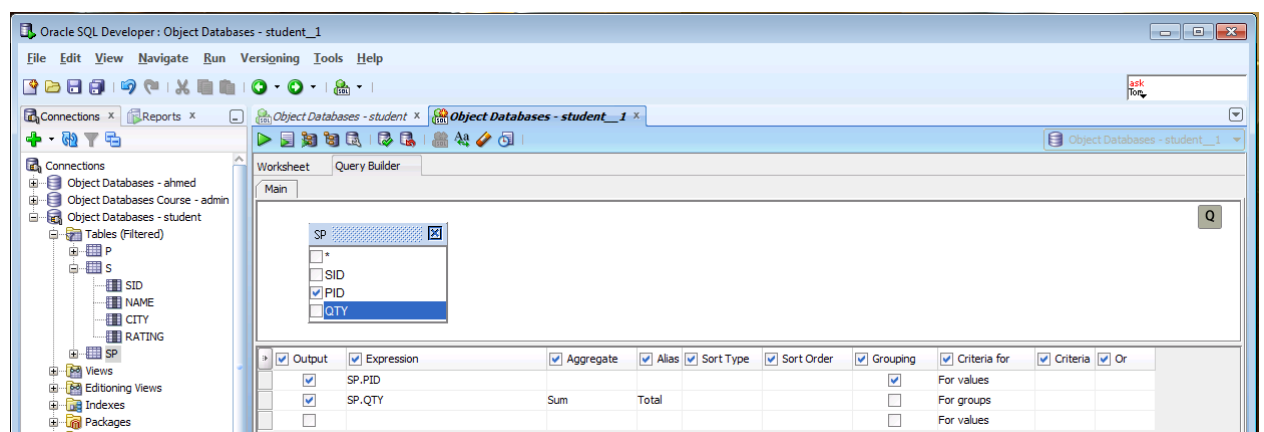
3. Proceeding similarly as above run the query and verify its results.
4. Modify the query to get a number of suppliers in Chelyabinsk (a resulting table still has one row and two columns: the first column should contain a string 'Number of suppliers in Chelyabinsk' and the second column should contain a number).
5. You are to write a query that gets minimum, average and maximum price of parts in the P table (columns should be named MinPrice, AvgPrice, MaxPrice).  
Proceeding similarly as above create a query using MIN, AVG, MAX aggregation functions for the Price column, give appropriate names to resulting columns. Run the query and verify its results.  
Modify the query to get statistics about black or white parts with weight more than 10. Run the query and verify its results.

**ASK INSTRUCTOR** to verify the results of your activity.

### Activity 16. Construct Simple Select Query with Grouping

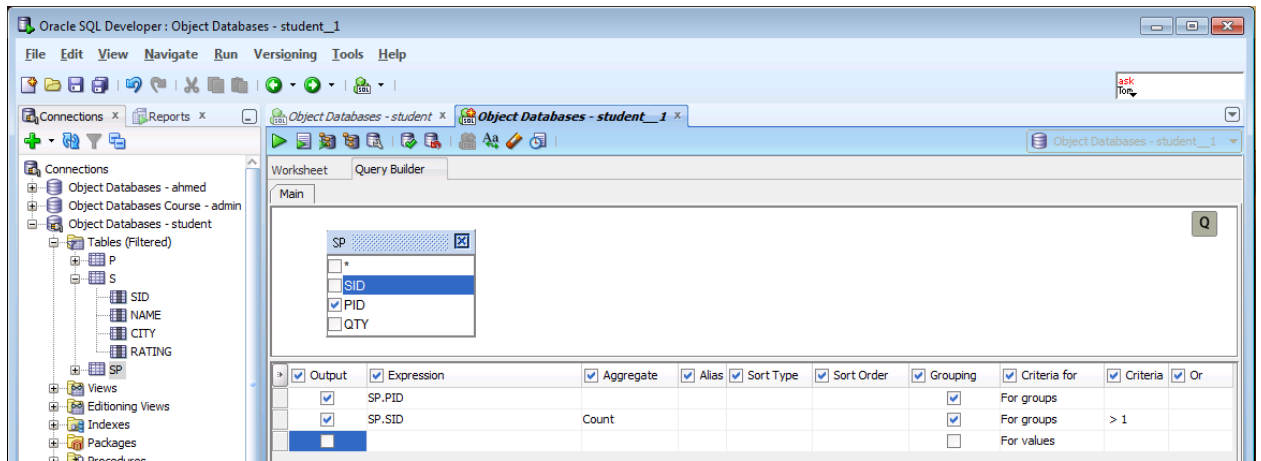
*The aim of this activity is to create a simple query using GROUP BY keyword.*

1. You are to write a query that gets an ID and quantity of supplies for every part.  
Proceeding similarly as above create a query as shown at the Fig. 22



**Fig. 22. Query with GROUP BY keyword**

2. Switch to Worksheet to verify that pure SQL code has the GROUP BY keyword. Run the query and verify its results.
3. You are to write a query that gets an ID of parts that supplied by more than one supplier.  
Proceeding similarly as above create a query as shown at the Fig. 23



**Fig. 23. Query with GROUP BY and HAVING keywords**

Switch to Worksheet to verify that pure SQL code has the GROUP BY and HAVING keywords. Run the query and verify its results.

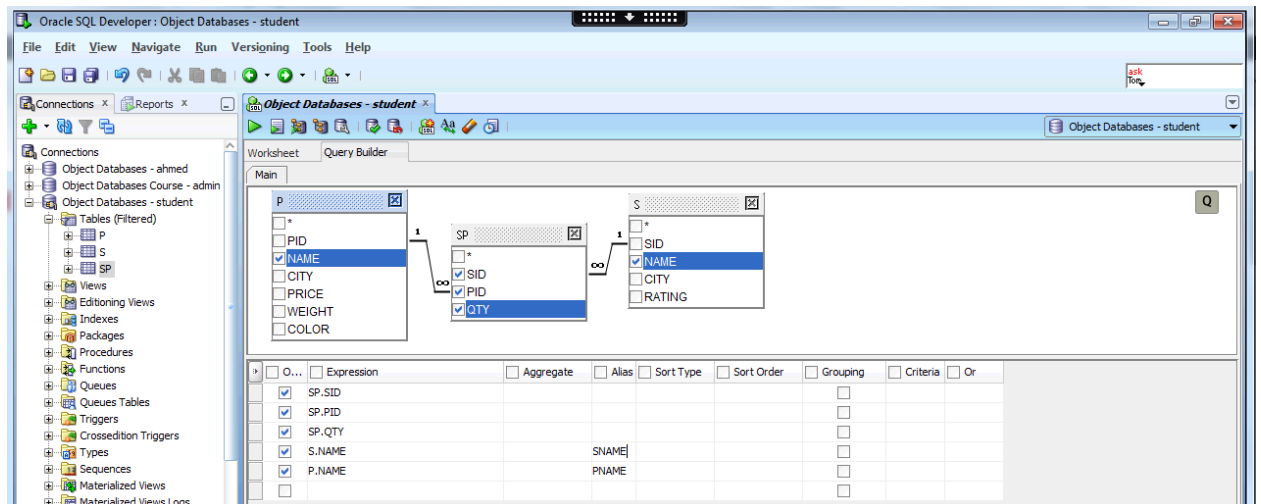
**ASK INSTRUCTOR** to verify the results of your activity.

### Activity 17. Construct Complex Query

*The aim of this activity is to create a complex queries (comprising more than one table).*

1. You are to write a query that gets all the supplies, additionally showing parts' names and suppliers' names.

Proceeding similarly as above create a query and drag tables P, SP, P into the Query Builder tab as shown at the Fig. 24.



**Fig. 24. Complex query with 3 tables**

Run the query and verify its results.

Switch to Worksheet tab to verify pure SQL code.

2. You are to write a query that gets names of suppliers that supply the P1002 part (without duplication).

3. You are to write a query that gets names of parts that supplied by the S0010 supplier (without duplication).
4. You are to write a query that gets names of parts from Chelyabinsk or Moscow that supplied by suppliers not from Chelyabinsk or Moscow (without duplication).
5. You are to write a query that gets names of suppliers with rating more than 10 and not from Chelyabinsk or Moscow that supply more than 100 black or red parts (without duplication).
6. You are to write a query that gets names of suppliers that have at least one supply (without duplication).
7. You are to write a query that gets names of parts that have been supplied at least once (without duplication).
8. You are to write a query that for every city gets city and total number of suppliers that have at least one supply.
9. You are to write a query that for every city gets city and total number of parts that have been supplied at least once.
10. You are to write a query that for every part gets ID, part's name and total number of suppliers that supplies this part.
11. You are to write a query that for every supplier gets ID, supplier's name and total number of parts that supplied by this supplier.
12. You are to write a query that gets ID, name and total quantity of supplies for every part.
13. You are to write a query that gets ID, name and total quantity of supplies for every supplier.
14. You are to write a query that gets ID, name of parts that supplied by more than one supplier. (*Note: use results of Activity 16. Construct Simple Select Query with Grouping*)
15. You are to write a query that gets ID, name of suppliers that supply exactly one part.
16. You are to write a query that gets names of parts that have price more than price of *all* the parts from Chelyabinsk (without duplication).
17. You are to write a query that gets IDs, names and rating *by words* (see Lab Scenario) of suppliers.

**ASK INSTRUCTOR** to verify the results of your activity.

## Lab 5. Triggers

**OBJECTIVE.** In this lab you will learn how to manage *triggers* in Oracle SQL Developer.

*Trigger* is procedural code that is automatically executed in response to certain event on a particular table in a database (e.g. before/after insertion or ). The trigger is mostly used for maintaining complex integrity constraints on the database (i.e. constraints involving more than one table).

There are four main types of triggers:

- 1) ROW LEVEL trigger: this gets executed before or after any column value of a row changes;
- 2) COLUMN LEVEL trigger: this gets executed before or after the specified column changes;
- 3) FOR EACH ROW type: this trigger gets executed once for each row of the result set caused by insert/update/delete;
- 4) FOR EACH STATEMENT type: this trigger gets executed only once for the entire result set, but fires each time the statement is executed.

### Activity 18. Create and manage triggers

*The aim of this activity* is to create and test trigger on SP table. This trigger should provide the following integrity constraints: if some supplier has made a supply with more than 10000 parts then its rating should be increased by 2 points.

Note that an appropriate trigger should be associated with the SP table and be of *for each row* type because every new record inserted into this table or every updated record of the SP table (i.e. if Qty value was increased more than 10000) may cause aforesaid increasing of respective supplier's rating. Also note that an appropriate trigger should be executed *after* inserting or updating the SP table.

1. Click right mouse button on the Triggers branch of connection tree and choose the New Table command.
2. Fill out the Create trigger form as shown at the Fig. 25.
3. Choose DDL tab and fill out the form as shown at the Fig. 26.
4. Insert some record into the SP table where Qty is greater than 10000. Verify that Rating of the respective record of the S table is increased because of ChageRating trigger was automatically executed.
5. Proceeding similarly verify that ChageRating trigger will be automatically executed after updating Qty field of some existing record in the SP table.

**ASK INSTRUCTOR** to verify the results of your activity.

6. Proceeding similarly create and test trigger that will decrease supplier's rating by 2 if that supplier had made a supply with less than 10 parts.

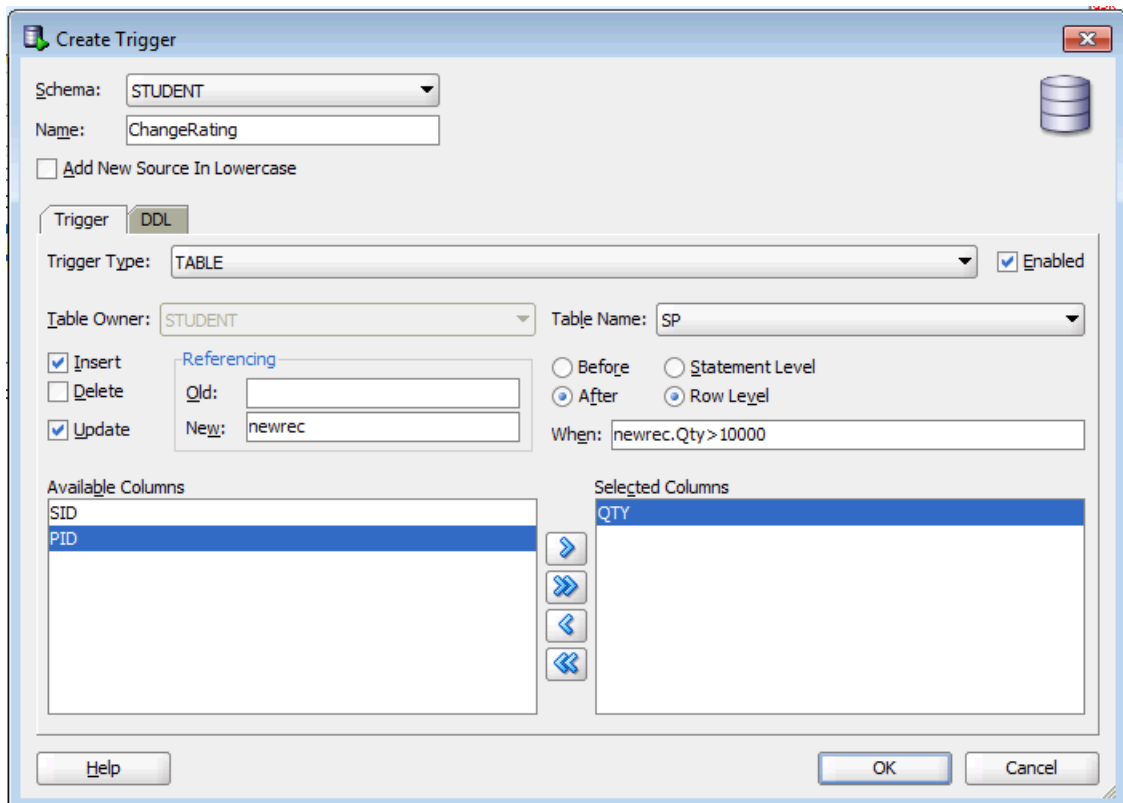


Fig. 25. Creating a trigger's interface

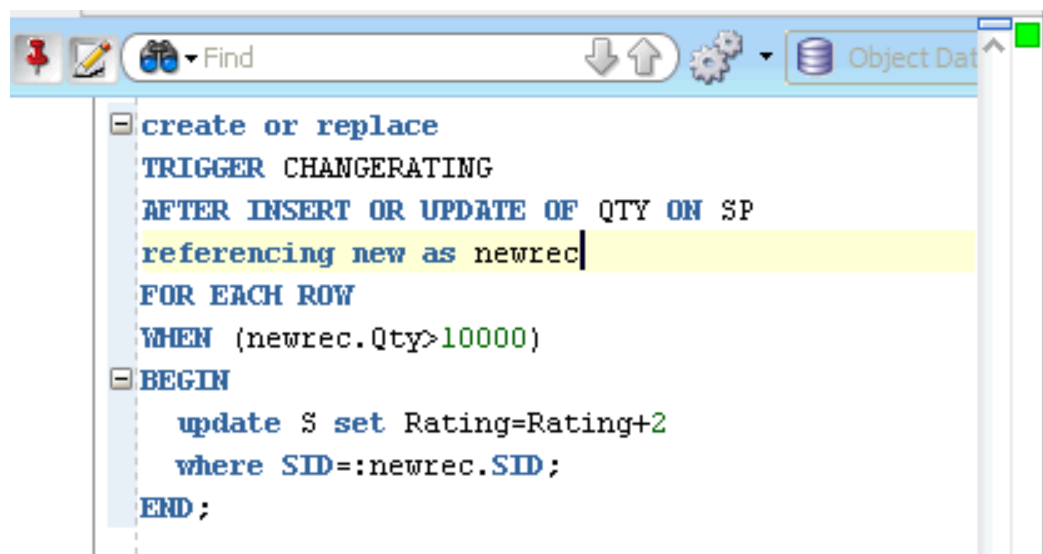


Fig. 26. Creating a trigger's implementation

**ASK INSTRUCTOR** to verify the results of your activity.



## Lab 6. Privileges and roles

**OBJECTIVE.** In this lab you will learn how to manage *privileges* and *roles* in Oracle SQL Developer.

A *privilege* is a right to execute a particular type of SQL statement or to access another user's object. Some examples of privileges include the right to connect to the database (create a session), create a table, select rows from another user's table, execute another user's stored procedure, etc. A *role* is a name for a set of privileges.

A privilege could be *granted* to or *revoked* from a user or a role.

### Activity 19. Granting and revoking privileges to users

*The aim of this activity* is to get an experience in managing privileges. You are to get access to tables of other users and open access to your tables to other users.

1. Run the following command in Query Builder window:  
`grant select, insert, update, delete on S to user1`  
where `user1` is login name of some your group mate.
2. Proceeding similarly get the privileges to select and modify data in the S table to `user2` (where `user2` is login name of some other your group mate).
3. Ask your group mates with login names `user1` and `user2` to open access to their S and P tables in the same way.
4. Insert one new record into the S and P tables of your group mates. Note that you should use `user1.S` and `user2.P` to indicate tables of your group mates.
5. Proceeding similarly update one record in the S and P tables of your group mates.
6. Proceeding similarly select all the records in the S and P tables of your group mates. Make sure that data in the S and P tables of your group mates was really changed.
7. Proceeding similarly make sure that data in your S and P tables was really changed by your group mates.

**ASK INSTRUCTOR** to verify the results of your activity.

8. Run the following command in Query Builder window:  
`revoke select, insert, update, delete on S from user1`
9. Proceeding similarly take away the privileges of `user1`.
10. Ask your group mates with `user1` and `user2` login names to check if they can access data in your S and P tables. Make sure that they can not.
11. Make sure that you lose the privileges to access data in `user1.S` and `user2.P` tables after their same manipulations.

## Activity 20. Managing roles

*The aim of this activity* is to get an experience in managing roles. You are to get access to tables of other users and open access to your tables to other users by means of roles.

1. Run the following command in Query Builder window:  
`create role reader`
2. Proceeding similarly as above grant select from the S, P and SP tables privileges to the reader role.
3. Proceeding similarly grant role `reader` to `user1`. Ask your group mate with login name `user1` to check if he/she can select but not modify data in your S, P and SP tables. Make sure that he/she can.

**ASK INSTRUCTOR** to verify the results of your activity.

4. Proceeding similarly create `writer` role with privileges to select, insert, update and delete data in the S, P and SP tables to `user2`. Ask your group mate with login name `user2` to check if he/she can select and modify data in your S, P and SP tables. Make sure that he/she can.
5. Try to select and modify data in your S, P and SP tables of your group mate (he/she has selected you as `user2` before). Make sure that you can.

**ASK INSTRUCTOR** to verify the results of your activity.

6. Run the following command in Query Builder window:  
`drop role reader`
7. Ask your group mate with `user1` login name to check if he/she can not access data in your S, P and SP tables. Make sure that he/she can not.
8. Proceeding similarly delete `writer` role and check the consequences of this operation.

**ASK INSTRUCTOR** to verify the results of your activity.