

Lab 3

Chan Wen Kang

A22EC0146

Section 6 Lesson 6 Exercise 1: Retrieving Data Using SELECT

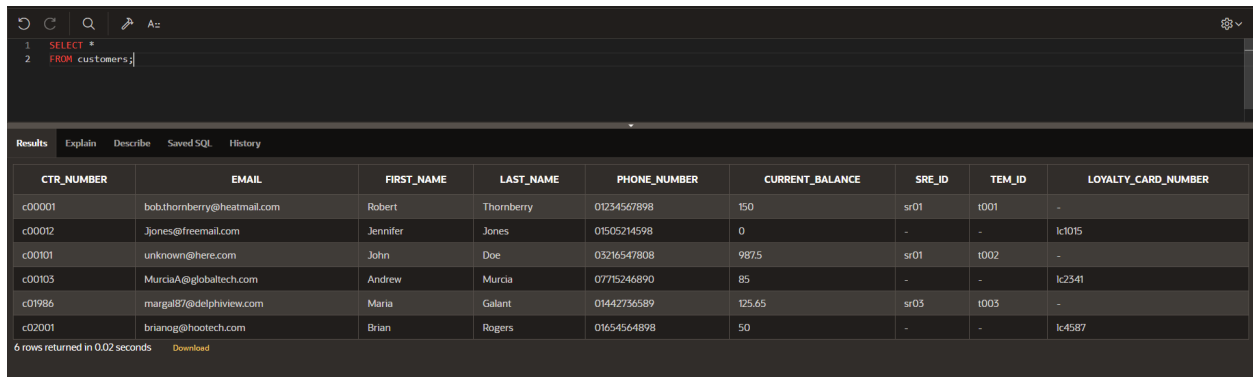
Write and Execute SELECT statements (S6L6 Objective 2)

In this exercise you will retrieve data that is stored in the database system by using a SELECT statement.

Part 1: Retrieving all columns from a table.

Using the SELECT * statement show all data stored in the following tables:

1. customers.

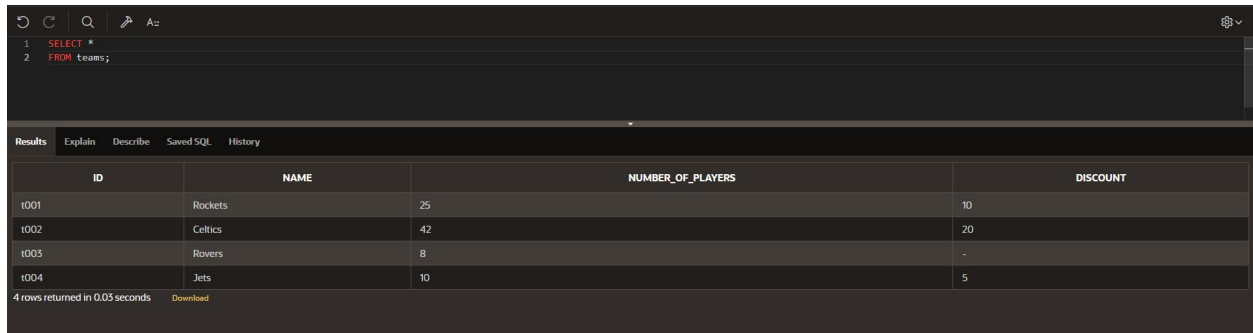


```
1 SELECT *
2 FROM customers;
```

CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID	TEM_ID	LOYALTY_CARD_NUMBER
c00001	bob.thornberry@heatmail.com	Robert	Thornberry	01234567898	150	sr01	t001	-
c00002	j.jones@freemail.com	Jennifer	Jones	01505214598	0	-	-	lc1015
c00101	unknown@here.com	John	Doe	03216547808	987.5	sr01	t002	-
c00103	MurciaA@globaltech.com	Andrew	Murcia	07715246890	85	-	-	lc2341
c01986	margal87@delphiview.com	Maria	Galant	01442736589	125.65	sr03	t003	-
c02001	brianog@hootech.com	Brian	Rogers	01654564898	50	-	-	lc4587

6 rows returned in 0.02 seconds [Download](#)

2. teams.

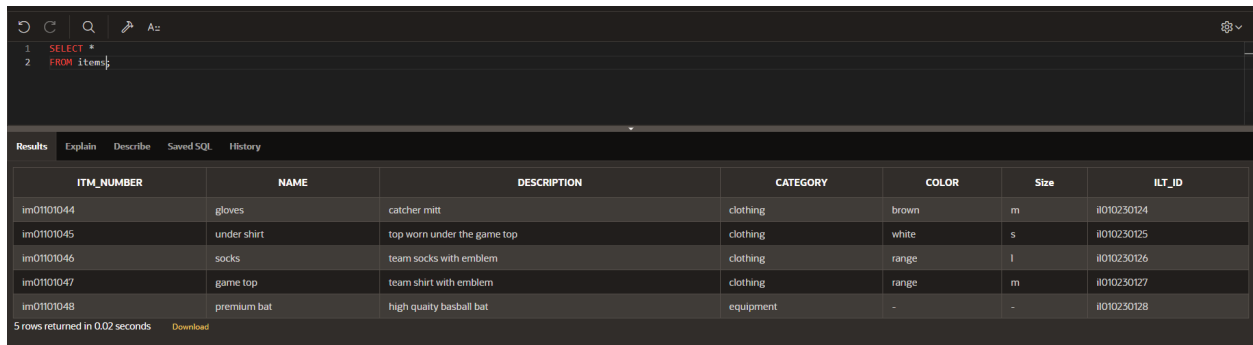


```
1 SELECT *
2 FROM teams;
```

ID	NAME	NUMBER_OF_PLAYERS	DISCOUNT
t001	Rockets	25	10
t002	Celtics	42	20
t003	Rovers	8	-
t004	Jets	10	5

4 rows returned in 0.05 seconds [Download](#)

3. items



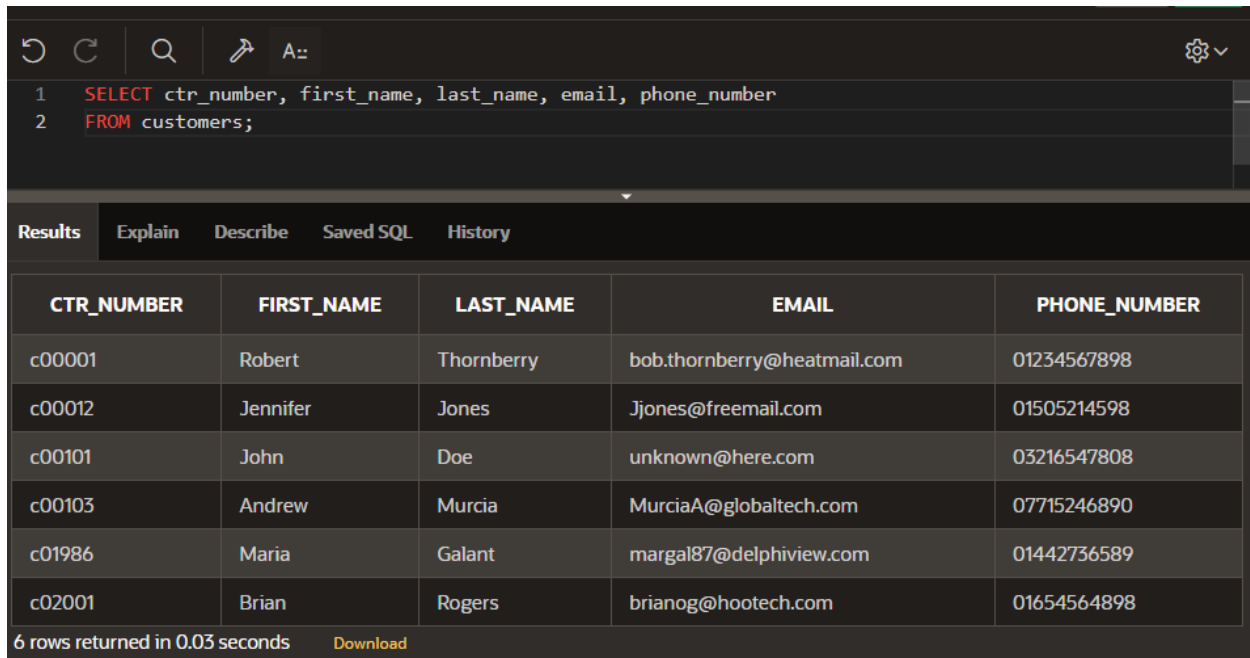
```
1 SELECT *
2 FROM items;
```

ITM_NUMBER	NAME	DESCRIPTION	CATEGORY	COLOR	Size	ILT_ID
im01101044	gloves	catcher mitt	clothing	brown	m	#010230124
im01101045	under shirt	top worn under the game top	clothing	white	s	#010230125
im01101046	socks	team socks with emblem	clothing	range	l	#010230126
im01101047	game top	team shirt with emblem	clothing	range	m	#010230127
im01101048	premium bat	high quality baseball bat	equipment	-	-	#010230128

5 rows returned in 0.02 seconds [Download](#)

Part 2: Selecting Specific Columns

1. Display the customer number, first name, last name, email and phone number of the customers.

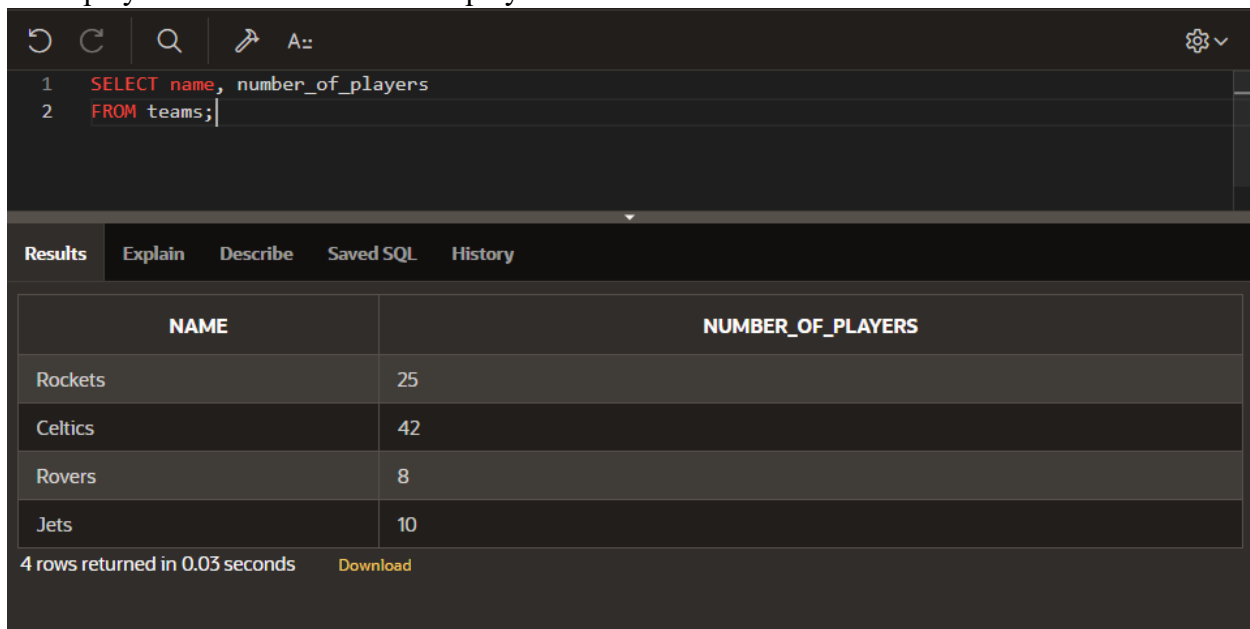


The screenshot shows a SQL query editor with a dark theme. The query is: `1 SELECT ctr_number, first_name, last_name, email, phone_number` and `2 FROM customers;`. Below the query, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying a table with 6 rows. The table has columns: CTR_NUMBER, FIRST_NAME, LAST_NAME, EMAIL, and PHONE_NUMBER. The data rows are: (c00001, Robert, Thornberry, bob.thornberry@heatmail.com, 01234567898), (c00012, Jennifer, Jones, Jjones@freemail.com, 01505214598), (c00101, John, Doe, unknown@here.com, 03216547808), (c00103, Andrew, Murcia, MurciaA@globaltech.com, 07715246890), (c01986, Maria, Galant, margal87@delphiview.com, 01442736589), and (c02001, Brian, Rogers, brianog@hootech.com, 01654564898). At the bottom, it says '6 rows returned in 0.03 seconds' and has a 'Download' button.

CTR_NUMBER	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER
c00001	Robert	Thornberry	bob.thornberry@heatmail.com	01234567898
c00012	Jennifer	Jones	Jjones@freemail.com	01505214598
c00101	John	Doe	unknown@here.com	03216547808
c00103	Andrew	Murcia	MurciaA@globaltech.com	07715246890
c01986	Maria	Galant	margal87@delphiview.com	01442736589
c02001	Brian	Rogers	brianog@hootech.com	01654564898

6 rows returned in 0.03 seconds [Download](#)

2. Display the name and number of players for each team.

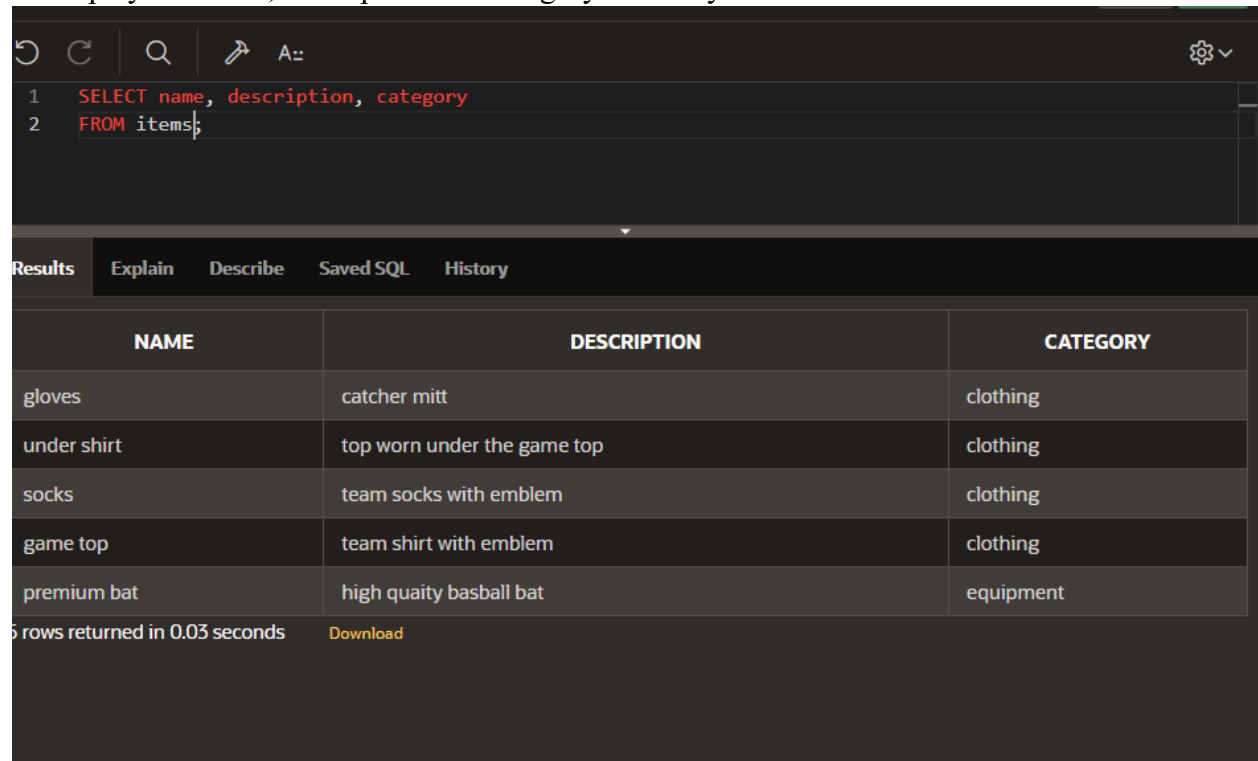


The screenshot shows a SQL query editor with a dark theme. The query is: `1 SELECT name, number_of_players` and `2 FROM teams;`. Below the query, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying a table with 4 rows. The table has columns: NAME and NUMBER_OF_PLAYERS. The data rows are: (Rockets, 25), (Celtics, 42), (Rovers, 8), and (Jets, 10). At the bottom, it says '4 rows returned in 0.03 seconds' and has a 'Download' button.

NAME	NUMBER_OF_PLAYERS
Rockets	25
Celtics	42
Rovers	8
Jets	10

4 rows returned in 0.03 seconds [Download](#)

3. Display the name, description and category for every item in the table.



The screenshot shows a SQL query editor with a dark theme. The query is: `1 SELECT name, description, category` and `2 FROM items;`. Below the editor is a tabbed interface with 'Results' selected. The results are displayed in a table with three columns: NAME, DESCRIPTION, and CATEGORY. The table contains five rows of data. At the bottom, it states '5 rows returned in 0.03 seconds' and has a 'Download' link.

```
1 SELECT name, description, category
2 FROM items;
```

NAME	DESCRIPTION	CATEGORY
gloves	catcher mitt	clothing
under shirt	top worn under the game top	clothing
socks	team socks with emblem	clothing
game top	team shirt with emblem	clothing
premium bat	high quaity baseball bat	equipment

5 rows returned in 0.03 seconds [Download](#)

2. Obl is considering giving a gift card to all its customers of 5.00 that can be used to reduce their current balance. Write a query that will show the customers first name, last name, customer

number, current balance and the value of their balance minus the gift value.

```
1 SELECT first_name, last_name, ctr_number, current_balance, (current_balance - 5.00) AS balance_after_gift
2 FROM customers;
```

Results Explain Describe Saved SQL History

FIRST_NAME	LAST_NAME	CTR_NUMBER	CURRENT_BALANCE	BALANCE_AFTER_GIFT
Robert	Thornberry	c00001	150	145
Jennifer	Jones	c00012	0	-5
John	Doe	c00101	987.5	982.5
Andrew	Murcia	c00103	85	80
Maria	Galant	c01986	125.65	120.65
Brian	Rogers	c02001	50	45

6 rows returned in 0.01 seconds [Download](#)

3. What would be the problem with implementing this scheme?

If the deduction exceeds the customer's current balance, it might result in negative balances, just like Jennifer on the above.

Part 2 : Using Column Aliases

1. You previously wrote a query that display the customer's first name, last name, current balance and monthly payment. Rewrite the query to use First Name, Last Name, Balance and Monthly Repayments as the column aliases. The aliases are to be shown exactly as described (case sensitive).

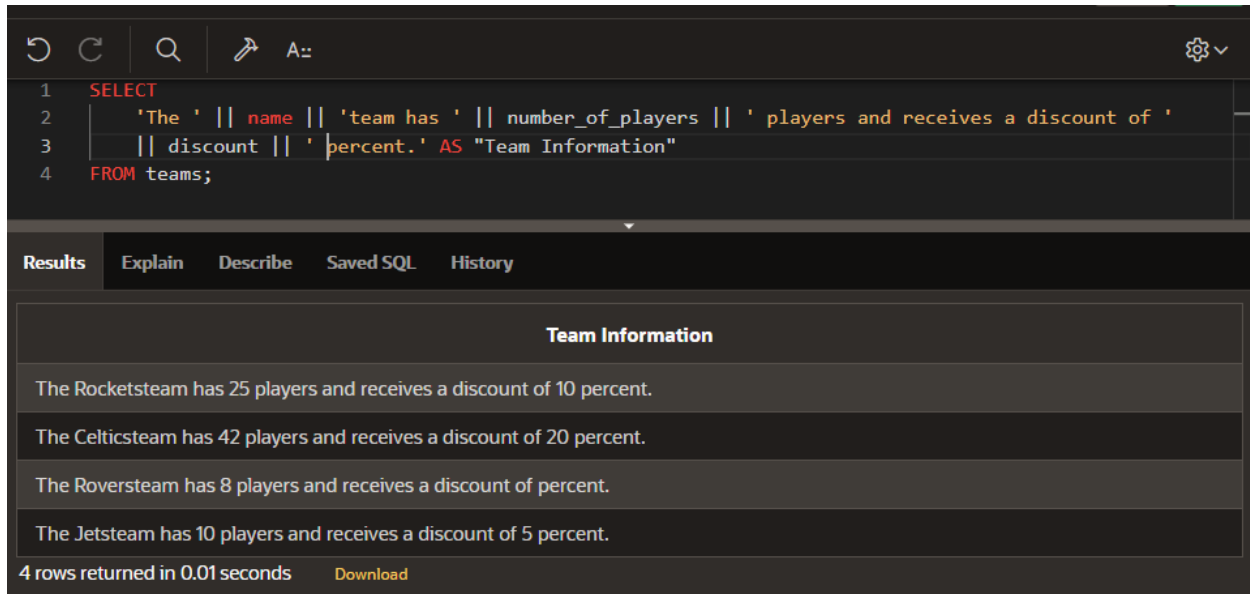
[illegible]

Part 3: Using Literal Character Strings

1. Write a query that will display the team information in the following format:

The Rockets team has 25 players and receives a discount of 10 percent.

Use Team Information as the column alias.



The screenshot shows a SQL IDE interface. The top bar contains icons for undo, redo, search, and a toolbar with a gear icon and a checkmark. The main editor area displays a SQL query:

```
1 SELECT
2     'The ' || name || 'team has ' || number_of_players || ' players and receives a discount of '
3     || discount || ' percent.' AS "Team Information"
4 FROM teams;
```

Below the editor, there is a tabbed interface with 'Results' selected. The results are displayed in a table with the column header 'Team Information'.

Team Information
The Rocketsteam has 25 players and receives a discount of 10 percent.
The Celticsteam has 42 players and receives a discount of 20 percent.
The Roversteam has 8 players and receives a discount of percent.
The Jetsteam has 10 players and receives a discount of 5 percent.

At the bottom of the results tab, it says '4 rows returned in 0.01 seconds' and there is a 'Download' button.

2. Why does the last team not show a discount?

The Rovers team does not show a discount because the discount column for the Rovers team in the teams table has NULL value. In SQL, a NULL value represents the absence of a value or an undefined value.

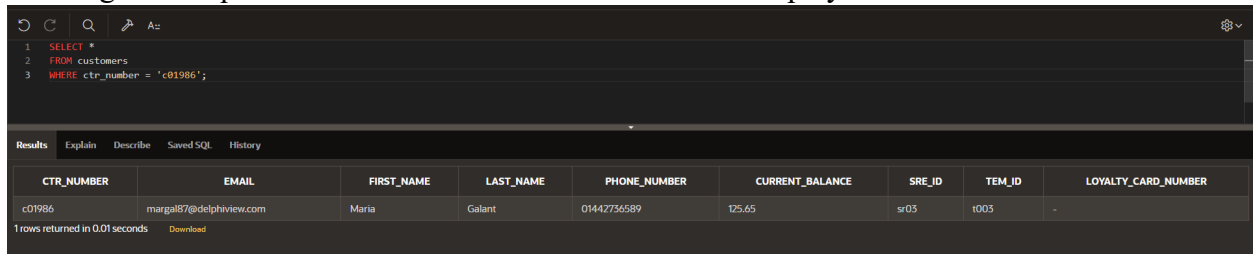
Section 6 Lesson 7 Exercise 1: Restricting Data Using WHERE

Limit rows using WHERE (S6L7 Objective 1)

In this exercise you will refine the data that is returned in your query by adding a WHERE clause to your SELECT statement.

Part 1: Using the WHERE Clause.

1. Using the unique customer number in the where clause display all columns for Maria Galant.



The screenshot shows a SQL IDE with a query editor and a results pane. The query editor contains the following SQL statement:

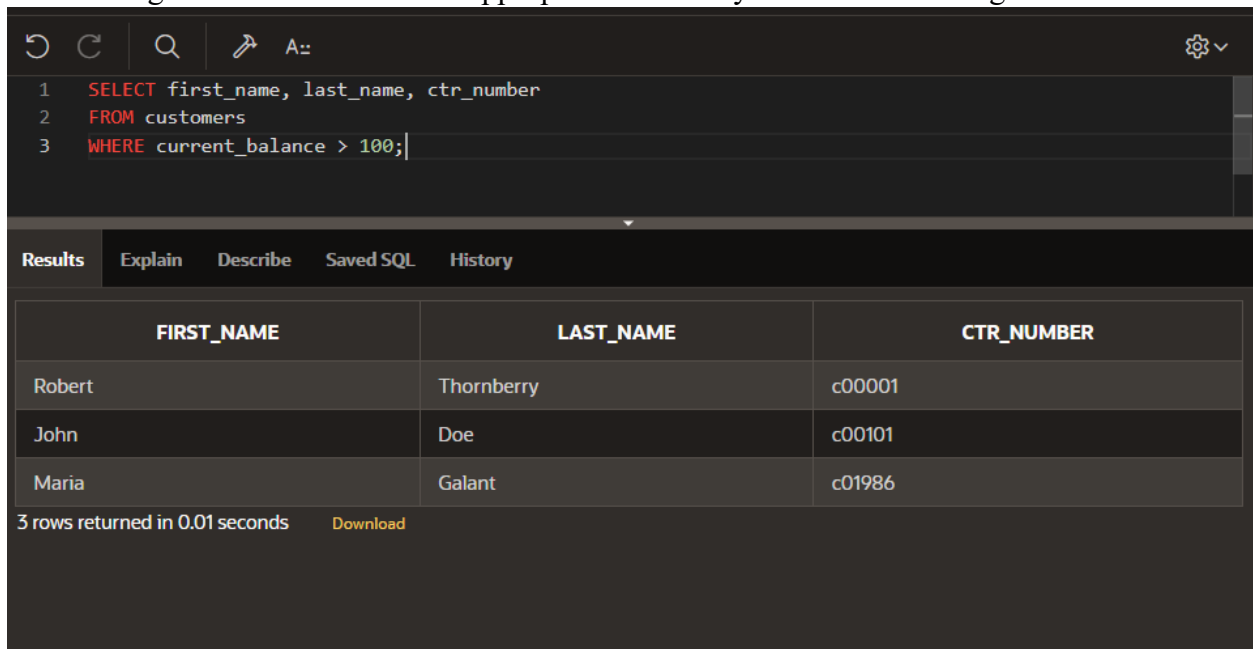
```
1 SELECT *
2 FROM customers
3 WHERE ctr_number = 'c01986';
```

The results pane displays a table with the following data:

CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID	TEM_ID	LOYALTY_CARD_NUMBER
c01986	margal87@delphiview.com	Maria	Galant	01442736589	125.65	sr03	t003	-

1 rows returned in 0.01 seconds [Download](#)

2. Display the first name, last name and customer number for all customers who have a current balance of greater than 100. Use an appropriate alias for your column headings.



The screenshot shows a SQL IDE with a query editor and a results pane. The query editor contains the following SQL statement:

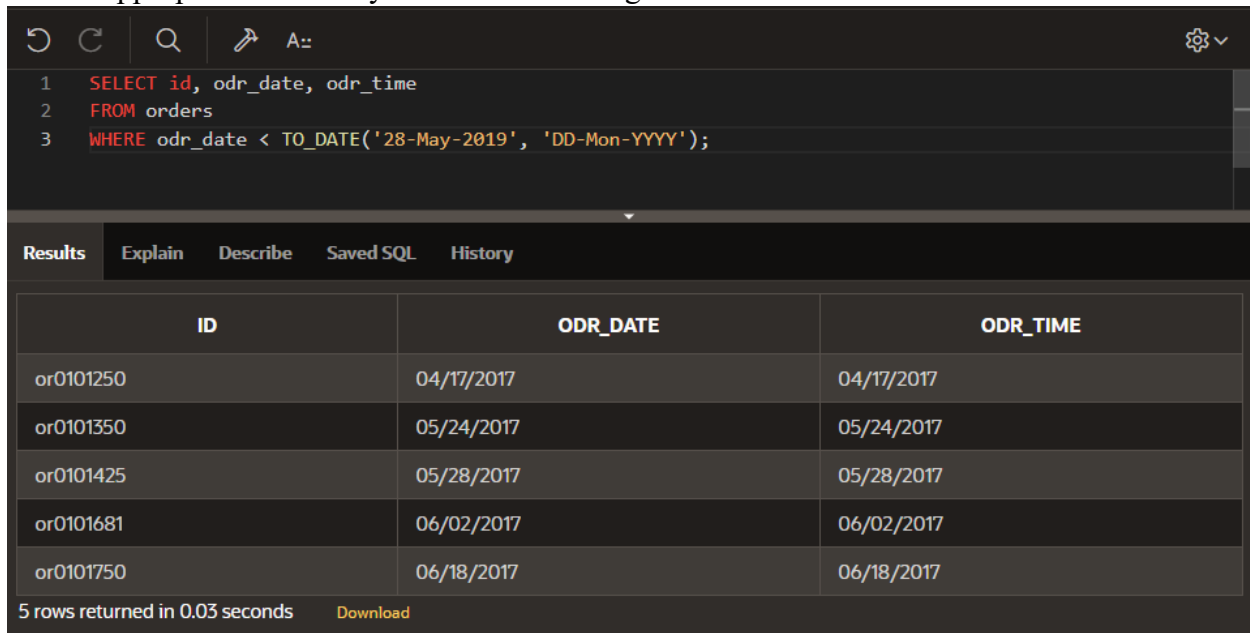
```
1 SELECT first_name, last_name, ctr_number
2 FROM customers
3 WHERE current_balance > 100;
```

The results pane displays a table with the following data:

FIRST_NAME	LAST_NAME	CTR_NUMBER
Robert	Thornberry	c00001
John	Doe	c00101
Maria	Galant	c01986

3 rows returned in 0.01 seconds [Download](#)

3. Display the order id, date and time of all orders that were placed before the 28th of May 2019. Use an appropriate alias for your column headings.



The screenshot shows a SQL query editor with the following query:

```
1 SELECT id, odr_date, odr_time
2 FROM orders
3 WHERE odr_date < TO_DATE('28-May-2019', 'DD-Mon-YYYY');
```

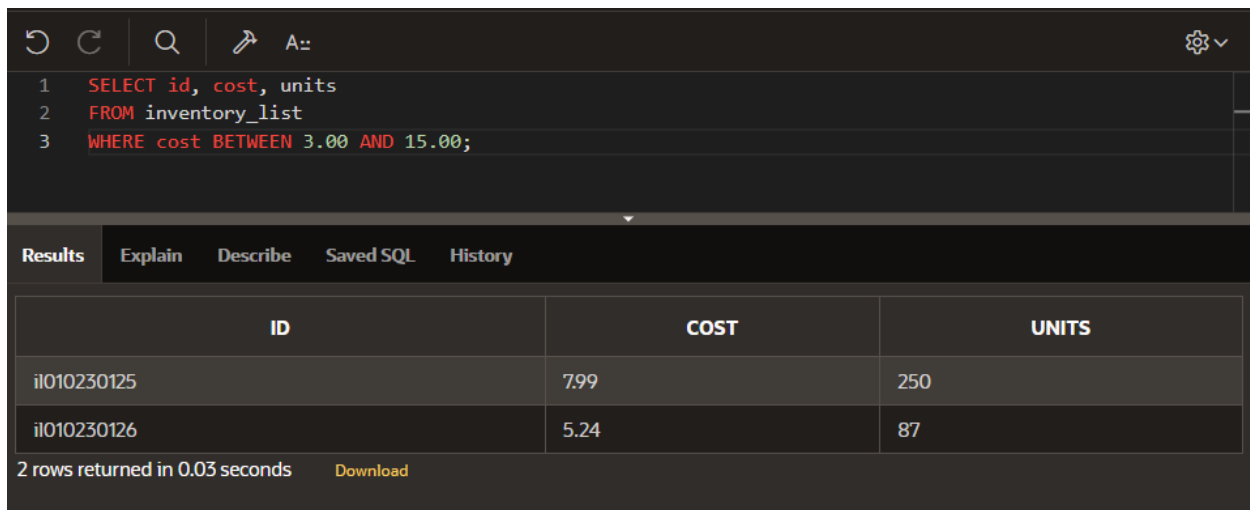
Below the query editor is a results table with the following data:

ID	ODR_DATE	ODR_TIME
or0101250	04/17/2017	04/17/2017
or0101350	05/24/2017	05/24/2017
or0101425	05/28/2017	05/28/2017
or0101681	06/02/2017	06/02/2017
or0101750	06/18/2017	06/18/2017

5 rows returned in 0.03 seconds [Download](#)

Part 2: Range Conditions: BETWEEN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that have a trade cost of between 3.00 and 15.00.



The screenshot shows a SQL query editor with the following query:

```
1 SELECT id, cost, units
2 FROM inventory_list
3 WHERE cost BETWEEN 3.00 AND 15.00;
```

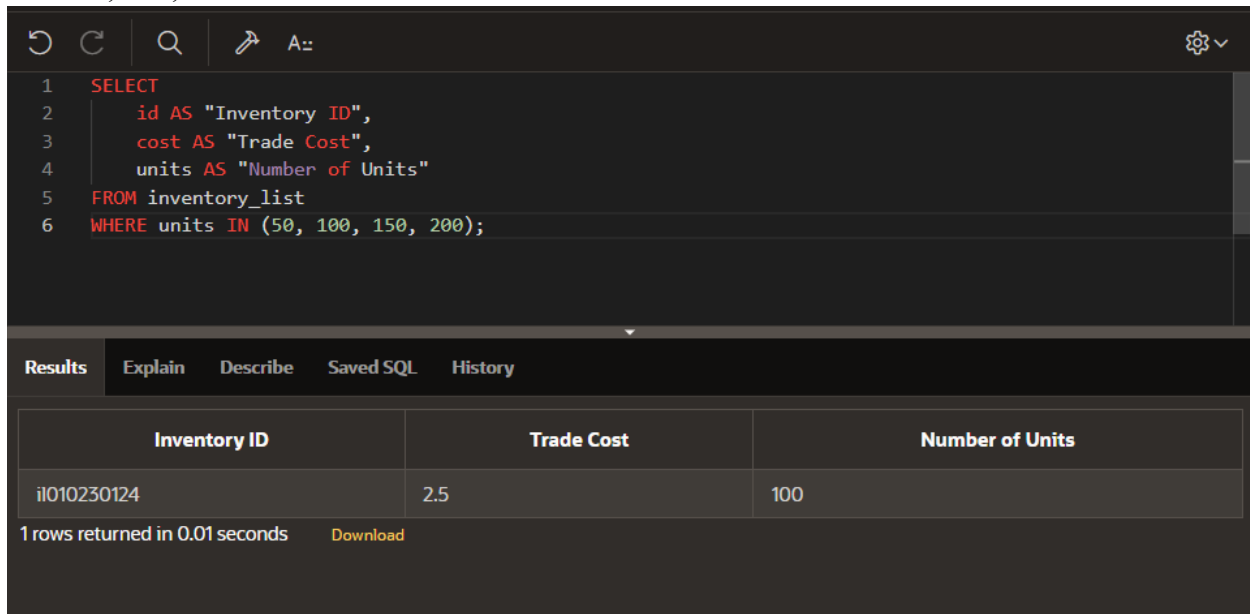
Below the query editor is a results table with the following data:

ID	COST	UNITS
il010230125	7.99	250
il010230126	5.24	87

2 rows returned in 0.03 seconds [Download](#)

Part 3: Membership Conditions: IN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that have 50, 100, 150 or 200 units in stock.



```
1 SELECT
2     id AS "Inventory ID",
3     cost AS "Trade Cost",
4     units AS "Number of Units"
5 FROM inventory_list
6 WHERE units IN (50, 100, 150, 200);
```

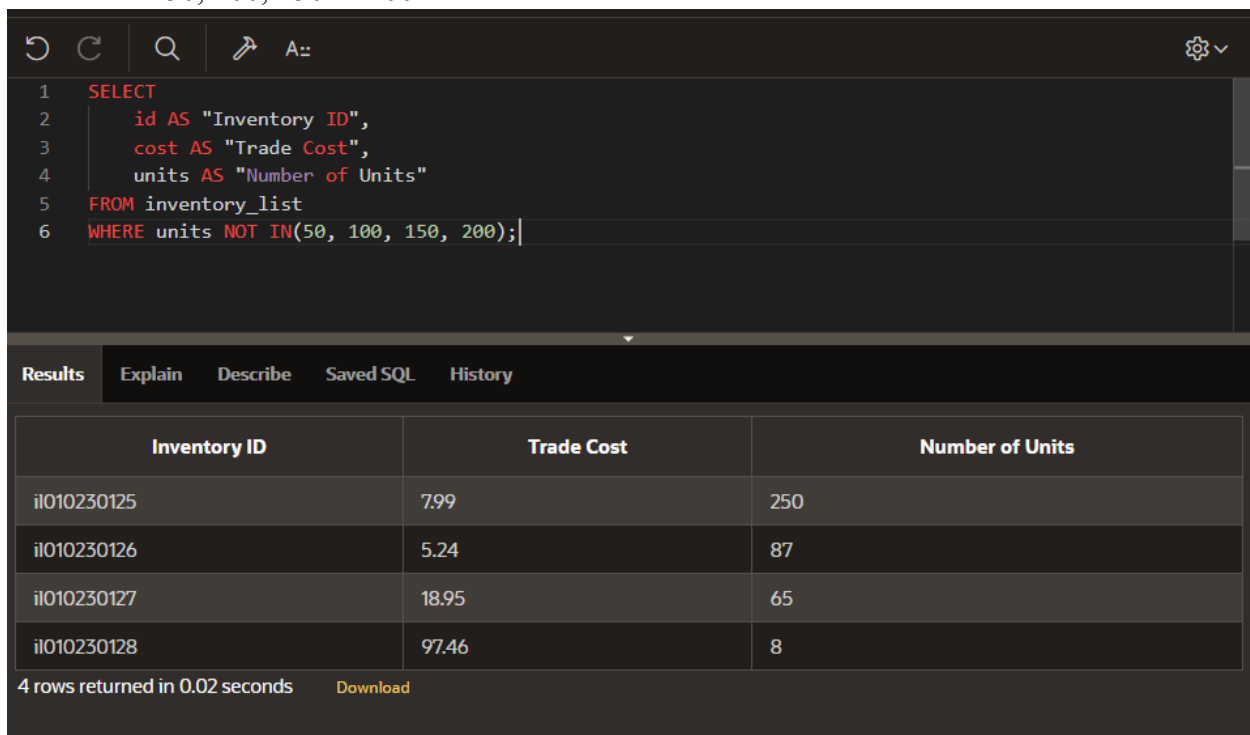
Results Explain Describe Saved SQL History

Inventory ID	Trade Cost	Number of Units
il010230124	2.5	100

1 rows returned in 0.01 seconds [Download](#)

Part 4: Membership Conditions: NOT IN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that do not have 50, 100, 150 or 200 units in stock.



```
1 SELECT
2     id AS "Inventory ID",
3     cost AS "Trade Cost",
4     units AS "Number of Units"
5 FROM inventory_list
6 WHERE units NOT IN(50, 100, 150, 200);
```

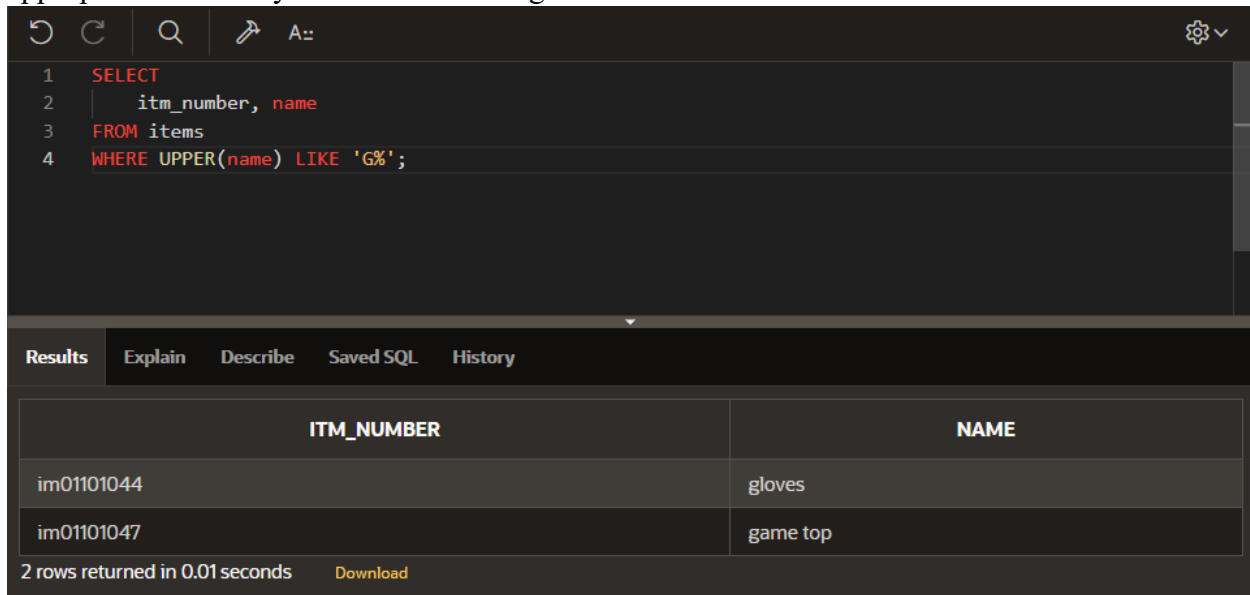
Results Explain Describe Saved SQL History

Inventory ID	Trade Cost	Number of Units
il010230125	7.99	250
il010230126	5.24	87
il010230127	18.95	65
il010230128	97.46	8

4 rows returned in 0.02 seconds [Download](#)

Part 5: Pattern Matching: LIKE Operator

1. Display item number and name of all items that have a name that begins with g. Use an appropriate alias for your column headings.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for undo, redo, search, and a settings icon. The SQL editor contains the following query:

```
1 SELECT
2   itm_number, name
3 FROM items
4 WHERE UPPER(name) LIKE 'G%';
```

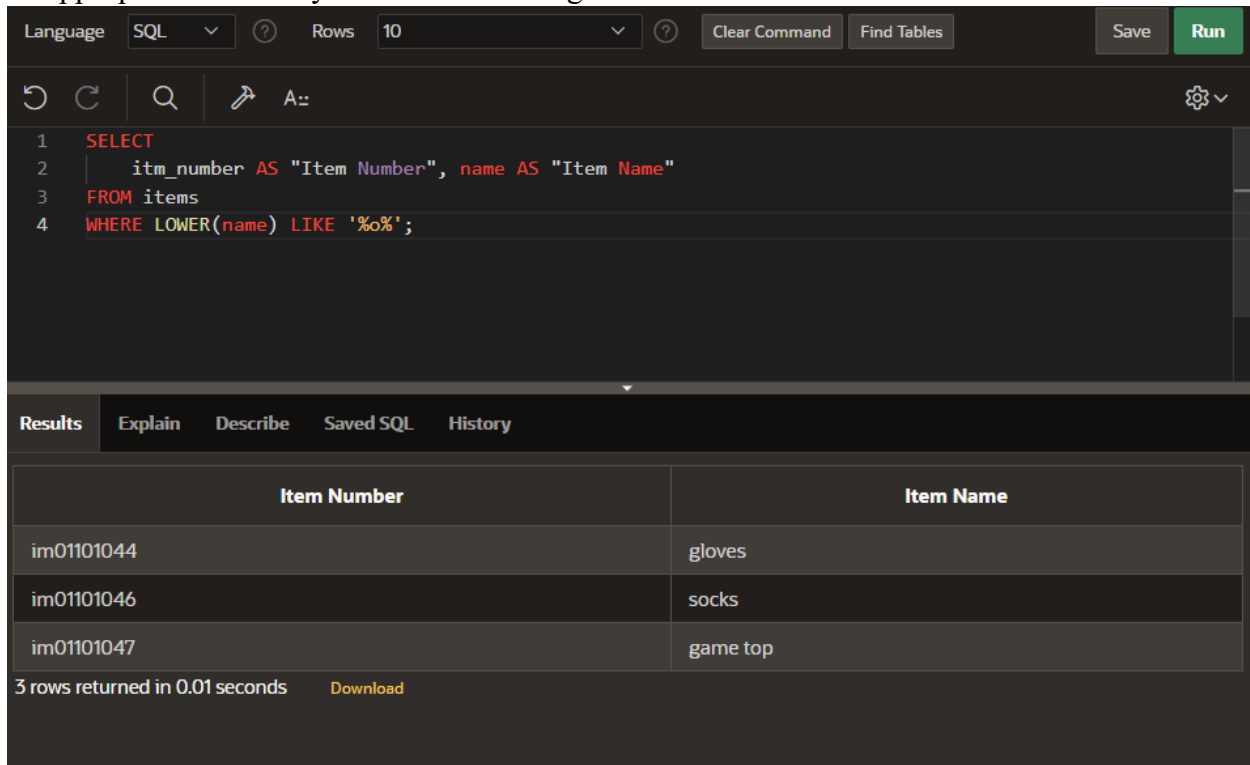
Below the editor, the 'Results' tab is active, showing a table with two columns: 'ITM_NUMBER' and 'NAME'. The table contains two rows of data.

ITM_NUMBER	NAME
im01101044	gloves
im01101047	game top

At the bottom, it states '2 rows returned in 0.01 seconds' and provides a 'Download' link.

Part 6 : Pattern Matching: Combining Wildcard Characters with the LIKE Operator

1. Display item number and name of all items that have a name that contain a lowercase o. Use an appropriate alias for your column headings.



The screenshot shows a SQL IDE interface. The top toolbar includes a 'Language' dropdown set to 'SQL', a 'Rows' dropdown set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL editor contains the following query:

```
1 SELECT
2   itm_number AS "Item Number", name AS "Item Name"
3 FROM items
4 WHERE LOWER(name) LIKE '%o%';
```

Below the editor, the 'Results' tab is active, showing a table with two columns: 'Item Number' and 'Item Name'. The table contains three rows of data.

Item Number	Item Name
im01101044	gloves
im01101046	socks
im01101047	game top

At the bottom, it states '3 rows returned in 0.01 seconds' and provides a 'Download' link.

Section 6 Lesson 7 Exercise 2: Restricting Data Using WHERE

Limit rows using WHERE (S6L7 Objective 1)

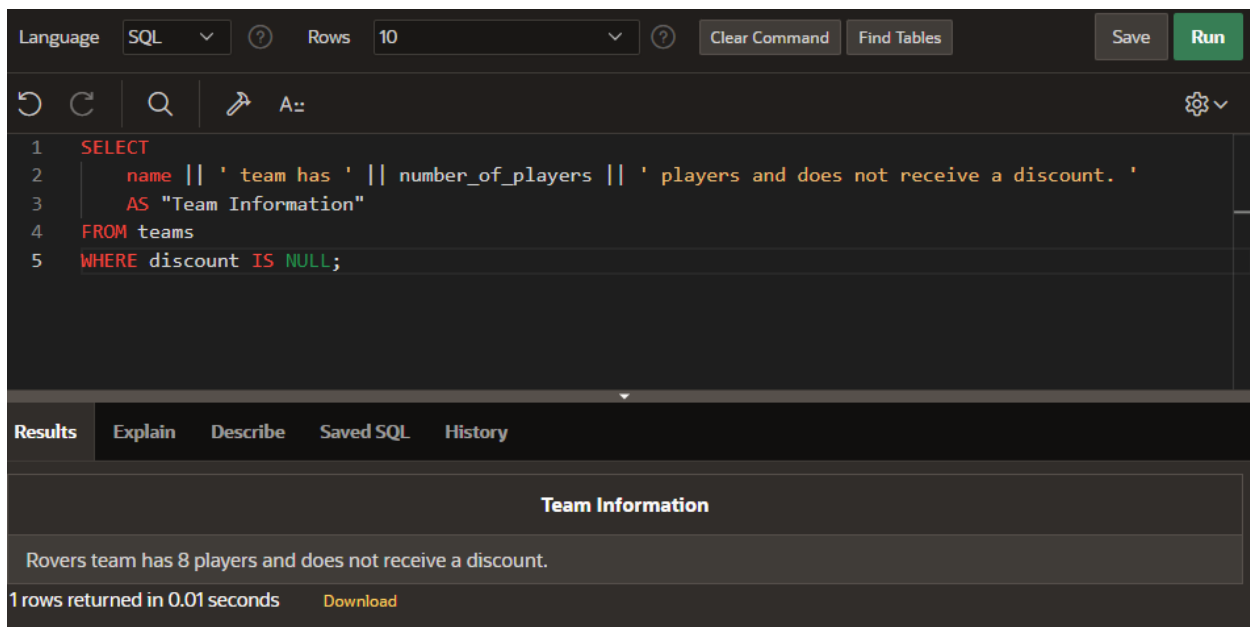
In this exercise you will refine the data that is returned in your query by adding a WHERE clause to your SELECT statement.

Part 1: Using the NULL Conditions

1. Write a query that will display information for teams that don't receive a discount in the following format:

The Rovers team has 25 players and does not receive a discount.

Use Team Information as the column alias.



The screenshot shows a SQL IDE interface. At the top, there are controls for 'Language' (set to SQL), 'Rows' (set to 10), and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below these is a toolbar with icons for undo, redo, search, and a command prompt. The main area contains a SQL query:

```
1 SELECT
2     name || ' team has ' || number_of_players || ' players and does not receive a discount. '
3     AS "Team Information"
4 FROM teams
5 WHERE discount IS NULL;
```

Below the query editor is a tabbed interface with 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a single row of data:

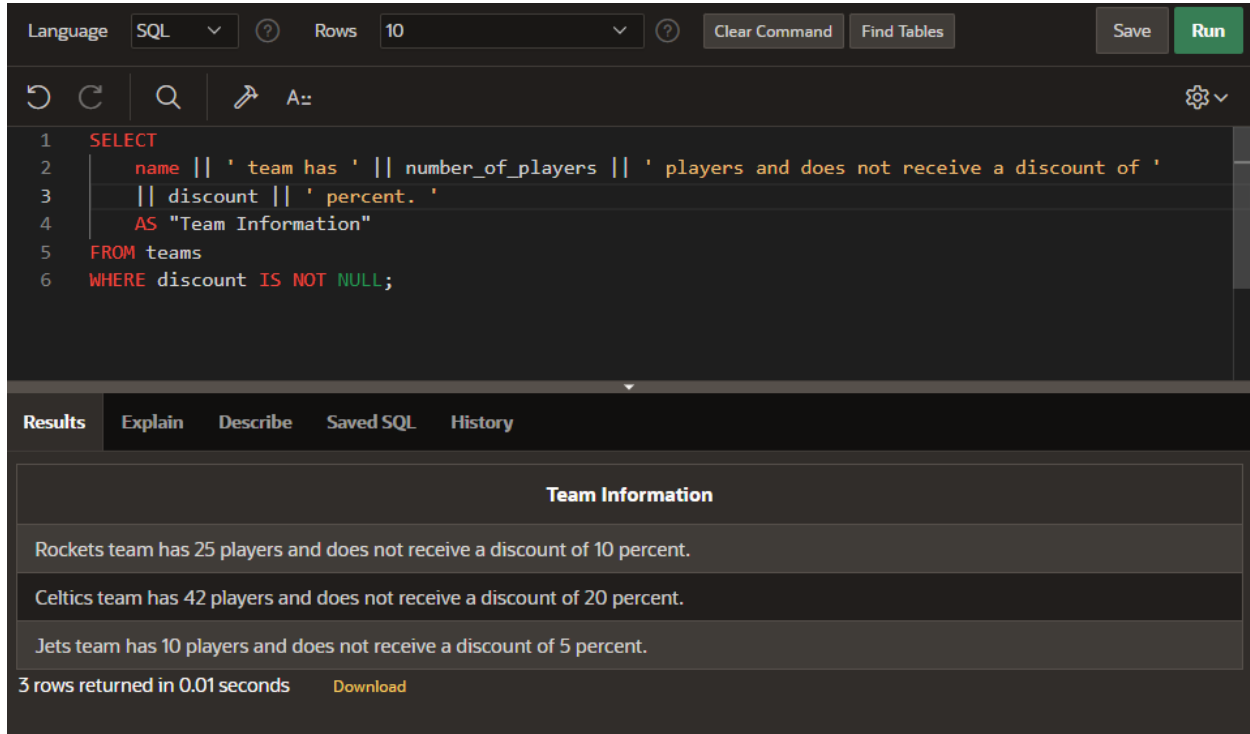
Team Information
Rovers team has 8 players and does not receive a discount.

At the bottom, it states '1 rows returned in 0.01 seconds' and provides a 'Download' link.

2. Write a query that will display information for only teams that receive a discount in the following format:

The Rockets team has 25 players and receives a discount of 10 percent.

Use Team Information as the column alias.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

```
1 SELECT
2     name || ' team has ' || number_of_players || ' players and does not receive a discount of '
3     || discount || ' percent. '
4     AS "Team Information"
5 FROM teams
6 WHERE discount IS NOT NULL;
```

Below the query editor is a 'Results' tab with sub-tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' sub-tab is active, showing a table with the title 'Team Information'. The table contains three rows of data:

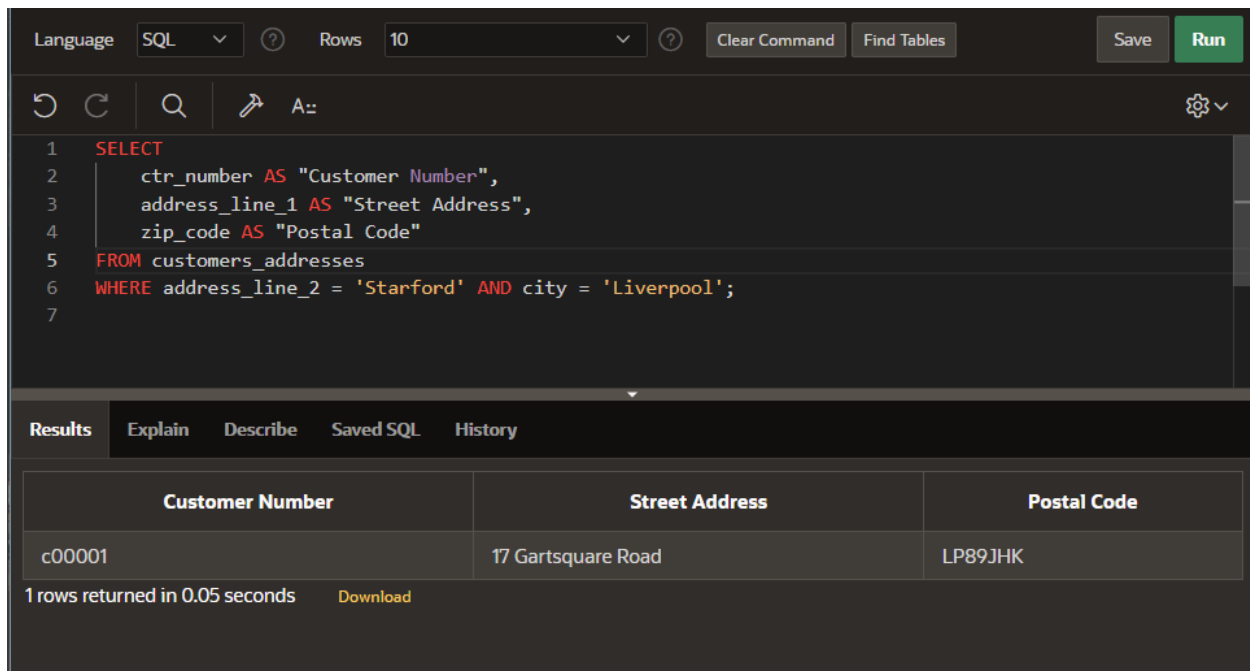
Team Information
Rockets team has 25 players and does not receive a discount of 10 percent.
Celtics team has 42 players and does not receive a discount of 20 percent.
Jets team has 10 players and does not receive a discount of 5 percent.

At the bottom of the results section, it says '3 rows returned in 0.01 seconds' and there is a 'Download' button.

Part 2: Logical Operators: AND

1. Write a query that will display the customer number, address line 1 and postal code for customers that live in the starford area of Liverpool. Use Customer Number, Street Address and

Postal Code as the column aliases.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

```
1 SELECT
2   ctr_number AS "Customer Number",
3   address_line_1 AS "Street Address",
4   zip_code AS "Postal Code"
5 FROM customers_addresses
6 WHERE address_line_2 = 'Starford' AND city = 'Liverpool';
7
```

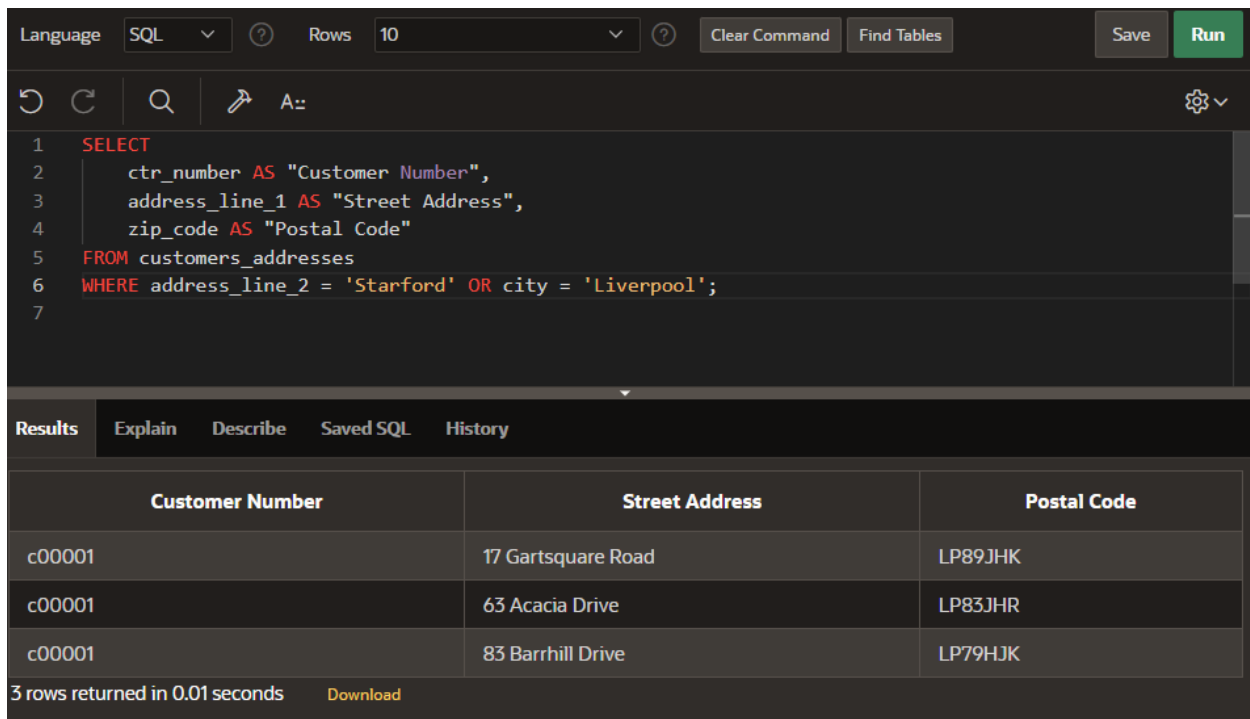
Below the query editor is a results panel with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with three columns: 'Customer Number', 'Street Address', and 'Postal Code'. The table contains one row of data.

Customer Number	Street Address	Postal Code
c00001	17 Gartsquare Road	LP89JHK

Below the table, it says '1 rows returned in 0.05 seconds' and there is a 'Download' button.

Part 3: Logical Operators: OR

1. Write a query that will display the customer number, address line 1 and postal code for customers that live in either starford or Liverpool in general. Use Customer Number, Street Address and Postal Code as the column aliases.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

```
1 SELECT
2   ctr_number AS "Customer Number",
3   address_line_1 AS "Street Address",
4   zip_code AS "Postal Code"
5 FROM customers_addresses
6 WHERE address_line_2 = 'Starford' OR city = 'Liverpool';
7
```

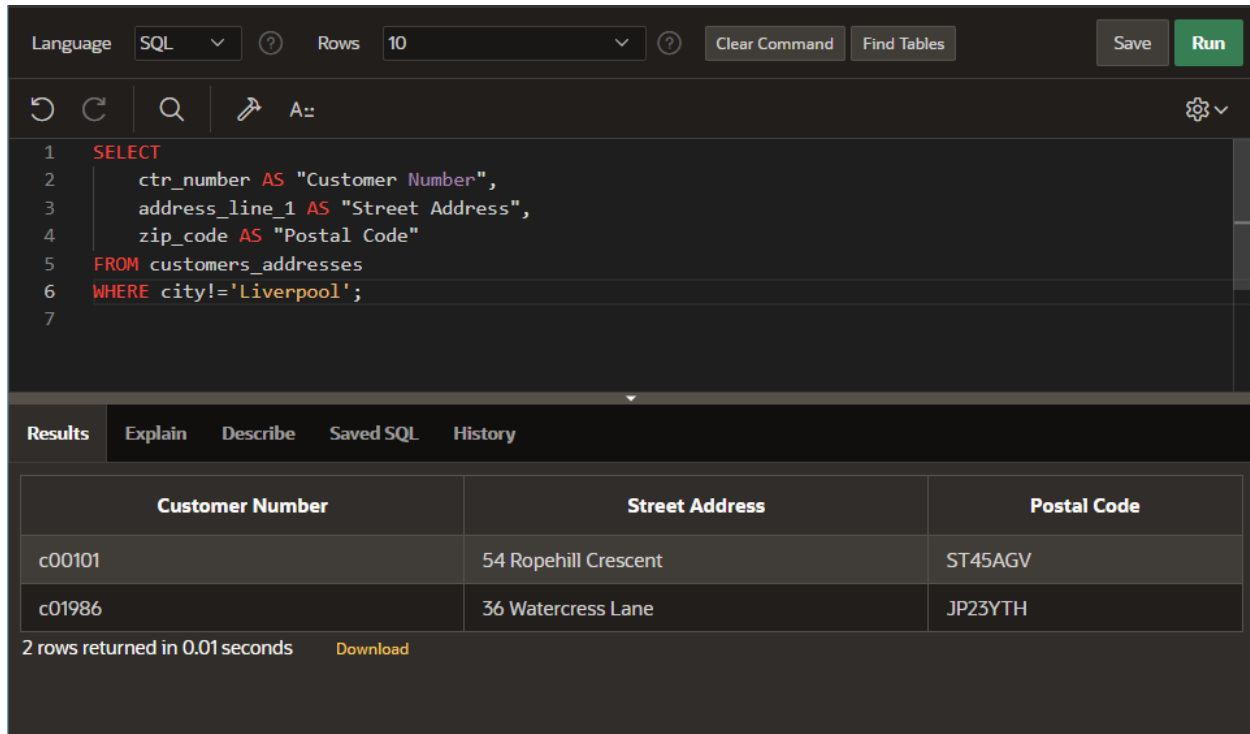
Below the query editor is a results panel with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with three columns: 'Customer Number', 'Street Address', and 'Postal Code'. The table contains three rows of data.

Customer Number	Street Address	Postal Code
c00001	17 Gartsquare Road	LP89JHK
c00001	63 Acacia Drive	LP83JHR
c00001	83 Barrhill Drive	LP79HJK

Below the table, it says '3 rows returned in 0.01 seconds' and there is a 'Download' button.

Part 4: Logical Operators: NOT Equal To

1. Write a query that will display the customer number, address line 1 and postal code for customers that do not live in Liverpool. Use Customer Number, Street Address and Postal Code as the column aliases.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

```
1 SELECT
2   ctr_number AS "Customer Number",
3   address_line_1 AS "Street Address",
4   zip_code AS "Postal Code"
5 FROM customers_addresses
6 WHERE city != 'Liverpool';
7
```

Below the query editor is a tabbed interface with 'Results' selected. The results are displayed in a table with three columns: 'Customer Number', 'Street Address', and 'Postal Code'. The table contains two rows of data. Below the table, it says '2 rows returned in 0.01 seconds' and there is a 'Download' button.

Customer Number	Street Address	Postal Code
c00101	54 Ropehill Crescent	ST45AGV
c01986	36 Watercress Lane	JP23YTH

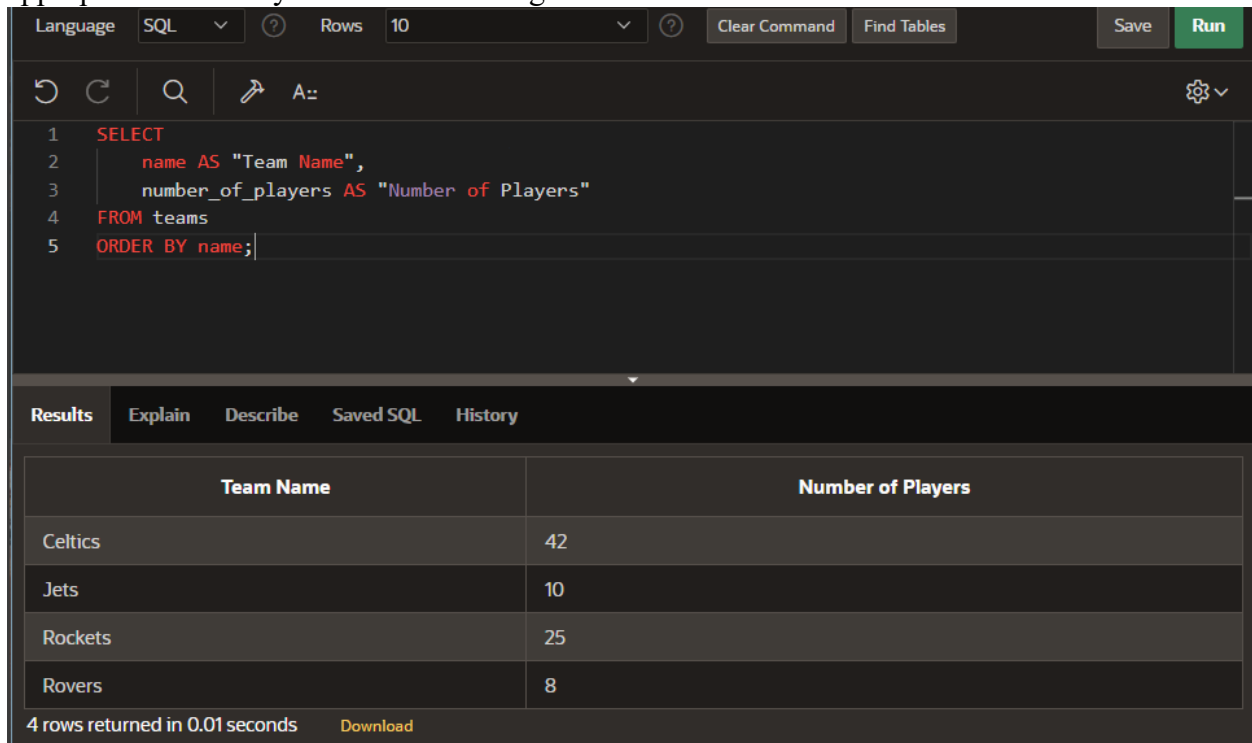
2 rows returned in 0.01 seconds [Download](#)

Section 6 Lesson 8 Exercise 1: Sorting Data Using ORDER BY

Use the ORDER BY Clause to Sort SQL Results (S6L8 Objective 1)

In this exercise you will sort the order of the data that is returned in your query by adding an ORDER BY clause to the end of your SELECT statement.

1. Display the team name and number of players alphabetically in order of team name. Use an appropriate alias for your column headings.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

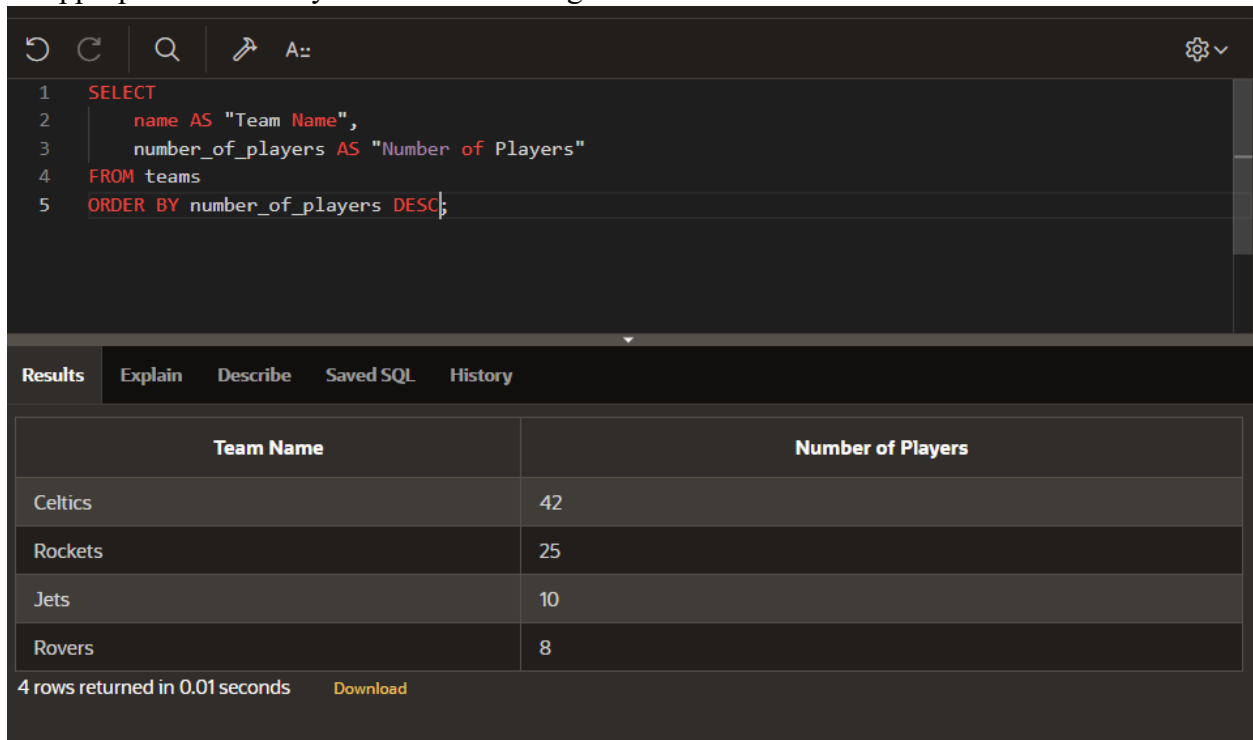
```
1 SELECT
2     name AS "Team Name",
3     number_of_players AS "Number of Players"
4 FROM teams
5 ORDER BY name;
```

Below the query editor is a results panel with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying a table with two columns: 'Team Name' and 'Number of Players'. The table contains four rows of data, sorted alphabetically by team name.

Team Name	Number of Players
Celtics	42
Jets	10
Rockets	25
Rovers	8

At the bottom of the results panel, it says '4 rows returned in 0.01 seconds' and there is a 'Download' button.

2. Display the team name and number of players in descending order of number of players. Use an appropriate alias for your column headings.



The screenshot shows a SQL query editor with the following query:

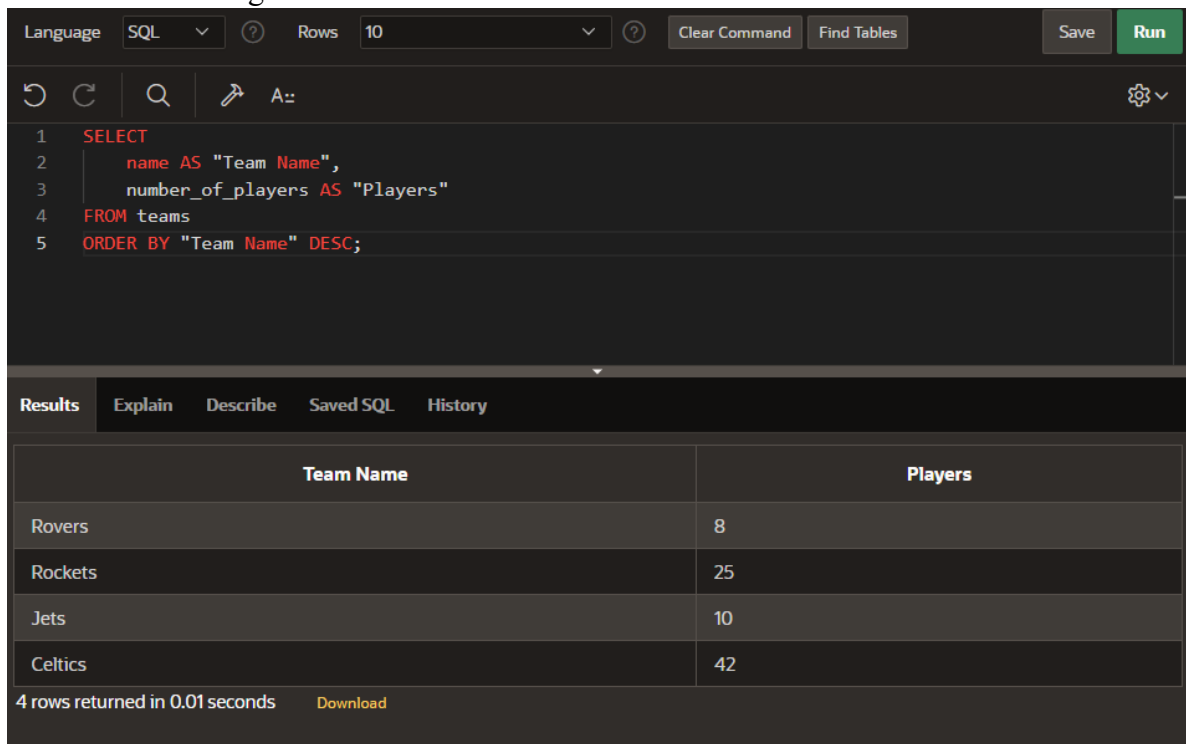
```
1 SELECT
2   name AS "Team Name",
3   number_of_players AS "Number of Players"
4 FROM teams
5 ORDER BY number_of_players DESC;
```

Below the query editor, the results are displayed in a table with two columns: "Team Name" and "Number of Players". The results are sorted in descending order of the number of players.

Team Name	Number of Players
Celtics	42
Rockets	25
Jets	10
Rovers	8

4 rows returned in 0.01 seconds [Download](#)

3. Display the team name and number of players alphabetically in order of team name. Use Team Name for the name alias and Players for the number of players. Sort the output in descending order of name using the alias in the ORDER BY clause.



The screenshot shows a SQL query editor with the following query:

```
1 SELECT
2   name AS "Team Name",
3   number_of_players AS "Players"
4 FROM teams
5 ORDER BY "Team Name" DESC;
```

Below the query editor, the results are displayed in a table with two columns: "Team Name" and "Players". The results are sorted in descending order of the team name.

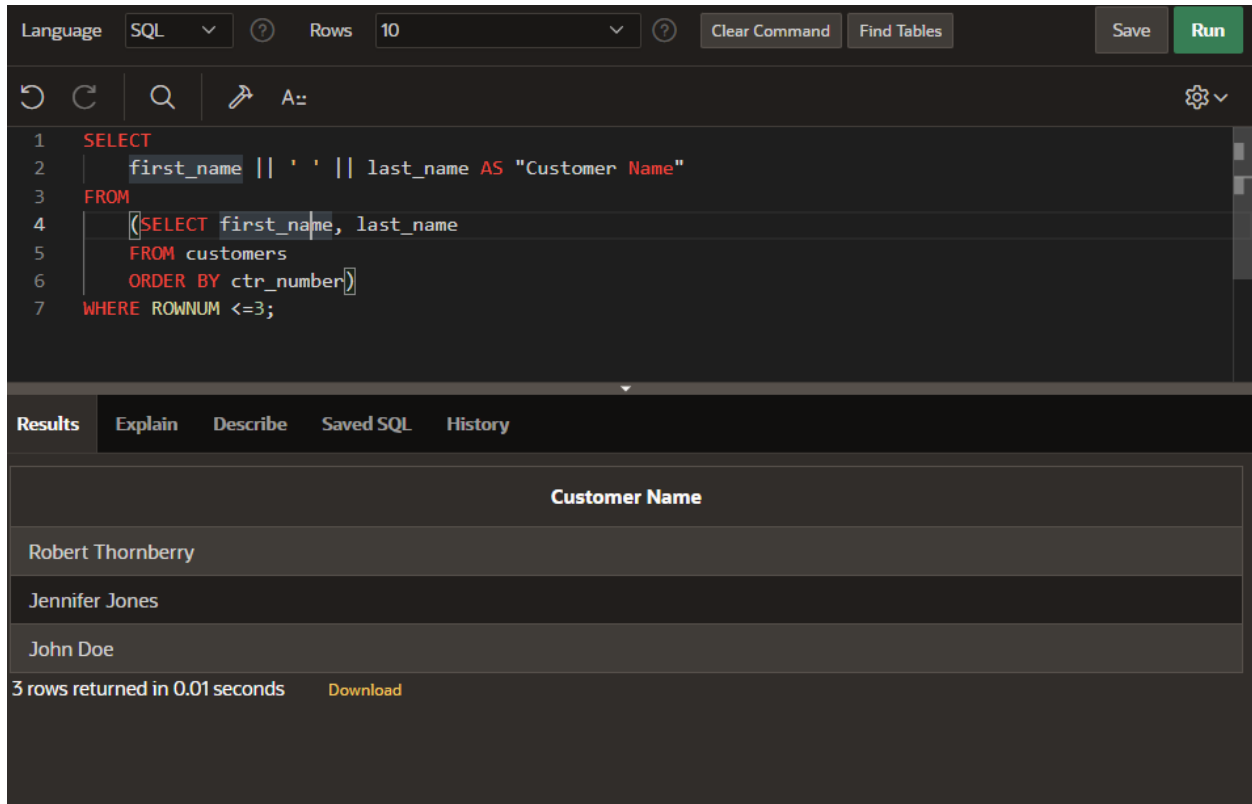
Team Name	Players
Rovers	8
Rockets	25
Jets	10
Celtics	42

4 rows returned in 0.01 seconds [Download](#)

Section 6 Lesson 8 Exercise 2: Sorting Data Using ORDER BY

Part 1 : TOP-N-ANALYSIS (S6L8 Objective 3)

1. The customers are numbered sequentially with each new customer being assigned a higher customer number. Use TOP-N-ANALYSIS to only show the First and last name of the first three customers. Show the customers first and last name in the same column using Customer Name as the column alias.



The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below the toolbar is a query editor with the following SQL code:

```
1 SELECT
2   first_name || ' ' || last_name AS "Customer Name"
3 FROM
4   (SELECT first_name, last_name
5    FROM customers
6    ORDER BY ctr_number)
7 WHERE ROWNUM <=3;
```

Below the query editor is a results pane with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with one column, 'Customer Name'. The table contains three rows of data:

Customer Name
Robert Thornberry
Jennifer Jones
John Doe

At the bottom of the results pane, it says '3 rows returned in 0.01 seconds' and there is a 'Download' button.

Part 2 : Using a Substitution Variable (S6L8 Objective 4)

1. Use a substitution variable that will allow you to enter the commission rate for the sales representatives. The first and last names should be displayed to screen for any sales representatives that earn that commission rate and the output should be ordered by their last name. Use an appropriate alias for your column headings

a. Code:

```
1  SELECT
2      first_name AS "First Name",
3      last_name AS "Last Name",
4      commission_rate AS "Commission Rate"
5  FROM sales_representatives
6  WHERE commission_rate = :CommissionRate
7  ORDER BY last_name;
```

b. Input

<div>Submit</div>	
Bind Variable	Value
:COMMISSIONRATE	<input type="text" value="5"/>

c. Result

Results

Explain

Describe

Saved SQL

History

First Name	Last Name	Commission Rate
Barry	Speed	5
Victoria	Wright	5

2 rows returned in 0.02 seconds

Download