



Instructor:

Ben Brumm

IT Software Consultant



Owner:

Azamat Nishonov

SQL Developer

Content

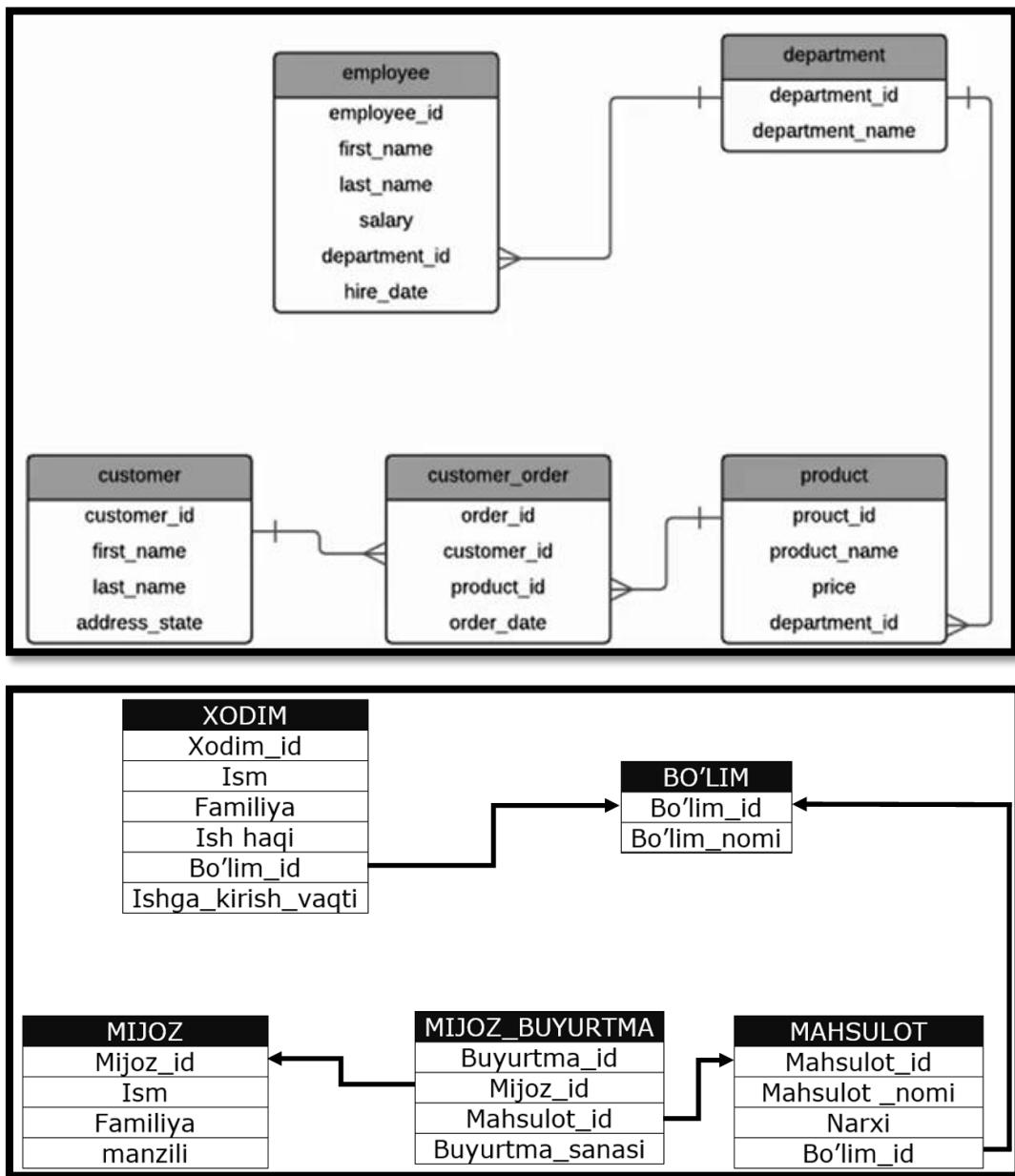
Sample Data	5
CREATE A NEW USER.....	6
VIEWING DATA.....	7
LIKE	7
AND, OR, NOT.....	7
NULL	8
DISTINCT.....	9
MORE OPERATORS	10
IN	10
BETWEEN	11
ALL keywords.....	11
ANY keywords.....	12
SORTING DATA.....	13
ORDER BY (1ta ustun uchun)	13
ORDER BY (bir nechta ustun uchun)	14
SET OPERATORS	14
UNION	14
INTERSECT.....	16
MINUS	17
MORE SETS	17
AGGREGATE FUNCTIONS AND GROUPING.....	19
COUNT()	19
Count data within GROUP BY	20
GROUP BY with ORDER BY	20
GROUP BY with HAVING	21
SUM function	21
MAX() and MIN() function	22
AVG function.....	23

JOINS.....	24
TABLE ALIASES	24
INNER JOIN	25
LEFT JOIN	27
RIGHT JOIN.....	28
FULL JOIN	30
NATURAL JOIN	31
CARTESIAN or CROSS JOIN.....	32
SELF JOIN	32
JOINING MANY TABLES	33
Alternative JOIN syntax	36
FUNCTIONS.....	39
String functions.....	39
Nesting functions within functions	40
Number Function	41
Date Function	42
Data Types and Conversion Function	43
Case Statement.....	44
Subqueries	46
About	46
Single row subquery	46
Multi row subquery	47
Inserting, Updating an Deleting Data.....	48
Inserting Data.....	48
Inserting Data From Another Table.....	50
Updating Data.....	50
Deleting Data.....	52
Commit and Rollback	53
Truncating Data	54

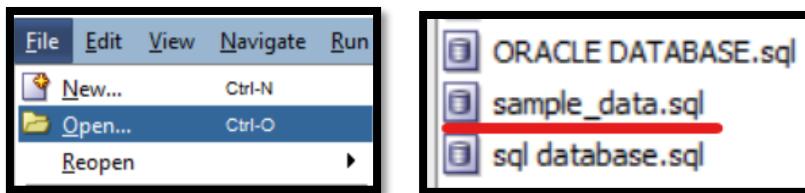
Creating, Altering and Dropping Tables.....	56
SQL Statement Types.....	56
Create a Table	56
Alter Table.....	57
Drop Table	59

Sample Data

Ma'lumotlar bazasi arxitekturasi



1. Ctrl + O (Open file)



2. Run Script [F5] ni bosamiz va baza o'rnatiladi.

A screenshot of the Oracle SQL Developer 'SQL Worksheet' tab. A green play button icon is highlighted with a red box. The worksheet area contains a single line of SQL: 'Insert into sample_data values (1);'. Below it, the 'Script Output' tab shows the message '1 row inserted.' indicating the success of the script execution.

CREATE A NEW USER

Sintaksis:

```
CREATE USER user_name
    IDENTIFIED BY password
    [DEFAULT TABLESPACE tablespace]
    [QUOTA {size | UNLIMITED} ON tablespace]
    [PROFILE profile]
    [PASSWORD EXPIRE]
    [ACCOUNT {LOCK | UNLOCK}];
```

Misol:

```
CREATE USER Azamat IDENTIFIED BY hayot;
```

Agar user_nameni qabul qilmasa unda siz faqat C## prefiks bilan foydalanuvchi yaratishingiz mumkin. Ya'ni:

```
CREATE USER c##Azamat IDENTIFIED BY hayot;
```

Boshqa misol:

```
CREATE USER intro_user IDENTIFIED BY mypassword;
GRANT CONNECT TO intro_user;
GRANT CREATE SESSION, GRANT ANY PRIVILEGE TO
intro_user;
GRANT UNLIMITED TABLESPACE TO intro_user;
GRANT CREATE TABLE TO intro_user;
```

VIEWING DATA

LIKE

Familiyasi Bu bilan boshlanuvchi shaxslar ma'lumotlari chiqariladi.

The screenshot shows two separate SQL queries in the SQL tab of Oracle SQL Developer. Both queries return results in the Query Result tab.

Left query:

```
SELECT first_name, last_name
FROM employee
WHERE first_name LIKE 'Ann_';
```

Right query:

```
SELECT first_name, last_name
FROM employee
WHERE last_name LIKE 'Bu%';
```

Both queries return 2 rows each:

	FIRST_NAME	LAST_NAME
1	Anna	Green
2	Anne	King

	FIRST_NAME	LAST_NAME
1	Sean	Burns
2	Lois	Butler

Sanalar bilan ishslash

The screenshot shows a single SQL query in the SQL tab of Oracle SQL Developer, which returns results in the Query Result tab.

```
SELECT first_name, hire_date
FROM employee
WHERE hire_date = '03.10.10';
```

The query returns 1 row:

	FIRST_NAME	HIRE_DATE
1	Sean	03.10.10

Ma'lumotlar formatini aniqlash

The screenshot shows a single SQL query in the SQL tab of Oracle SQL Developer, which returns results in the Query Result tab.

```
SELECT value
FROM SYS.nls_database_parameters
WHERE PARAMETER = 'NLS_DATE_FORMAT';
```

The query returns 1 row:

	VALUE
1	DD-MON-RR

AND, OR, NOT

1-holat. Bu yerda ikkovi alohida bajarilib yig'iladi.

1-shart. hire_date < '01.12.14' AND first_name = 'Barbara'

2-shart. salary < 70000

```
SELECT *
FROM employee
WHERE hire_date < '01.12.14'
AND first_name = 'Barbara'
OR salary < 70000
;
```

Query Result | Fetched 50 rows in 0,004 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	MANAGER_ID	HIRE_DATE
1	Michelle	Foster	48000	8	162	27.08.11
2	Carolyn	Hudson	47000	7	199	04.12.16
3	Patrick	Berry	51000	3	159	12.10.11

2-holat. Bu yerda 1-shart aniq bajariladi, 2-shartdan 2 tadan bittasi alohida alohida tanlanadi.

1-shart. hire_date < '01.12.14' AND

2-shart. first_name = 'Barbara' [yoki] salary < 70000

```
SELECT *
FROM employee
WHERE hire_date < '01.12.14'
AND (first_name = 'Barbara'
OR salary < 70000)
;
```

Query Result | Fetched 50 rows in 0,004 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	MANAGER_ID	HIRE_DATE
1	Michelle	Foster	48000	8	162	27.08.11
2	Patrick	Berry	51000	3	159	12.10.11
3	Jessica	Elliott	21000	7	70	02.07.10

NULL

NUL qiymatlarni aniqlash [Ochiq (bo'sh)]

```
SELECT first_name, salary  
FROM employee  
WHERE salary IS NULL  
;
```

Query Result x

All Rows Fetched: 2

	FIRST_NAME	SALARY
1	Anthony	(null)
2	Jennifer	(null)

NULL bo'limgan qiymatlarni aniqlash

```
SELECT first_name, salary  
FROM employee  
WHERE salary IS NOT NULL  
;
```

Query Result x

Fetched 50 rows in 0.00 sec

	FIRST_NAME	SALARY
1	Michelle	48000
2	Cheryl	79000
3	Carolyn	47000

DISTINCT

DISTINCT — farqli qiymatlarni chiqarish uchun ishlataladi. 2ta bir xil qiymatni faqat bittasini chiqaradi.

1-holat. Ismlarni chiqardi. (200 ta qiymat)

2-holat. Faqat farqli ismlarni chiqardi. (126 ta)

3-holat. Ism va familiyasi bir vaqtida farq qiluvchilarni chiqardi. (200 ta)

```
SELECT first_name
FROM employee;
```

Query Result

All Rows Fetched: 200

FIRST_NAME
1 Michelle
2 Cheryl
3 Carolyn

```
SELECT DISTINCT first_name
FROM employee;
```

Query Result

All Rows Fetched: 126

FIRST_NAME
1 Michelle
2 Cheryl
3 Carolyn

```
SELECT DISTINCT first_name, last_name
FROM employee;
```

Query Result

All Rows Fetched: 200 in 0,014 seconds

FIRST_NAME	LAST_NAME
1 Michelle	Foster
2 Cheryl	Turner
3 Carolyn	Hudson

MORE OPERATORS

IN

OR ni o'rnnini bosuvchi **IN**

```
SELECT employee_id, last_name
FROM employee
WHERE last_name = 'Foster'
OR last_name = 'Elliott'
OR last_name = 'Mitchell'
;
```

Query Result

All Rows Fetched: 6 in 0,000 seconds

EMPLOYEE_ID	LAST_NAME
1	1 Foster
2	6 Elliott
3	14 Mitchell
4	123 Elliott
5	138 Foster
6	149 Mitchell

```
SELECT employee_id, last_name
FROM employee
WHERE last_name IN ('Foster', 'Elliott', 'Mitchell')
;
```

Query Result

All Rows Fetched: 6 in 0,006 seconds

EMPLOYEE_ID	LAST_NAME
1	1 Foster
2	6 Elliott
3	14 Mitchell
4	123 Elliott
5	138 Foster
6	149 Mitchell

BETWEEN

BETWEEN bu chegaraviy oraliqdagi qiymatlar

```

SELECT first_name, hire_date
FROM employee
WHERE hire_date >= '01.05.2016'
AND hire_date <= '31.05.2016';
;
```

Query Result | All Rows Fetched: 3 in 0,00 seconds

FIRST_NAME	HIRE_DATE
1 John	25.05.16
2 Jacqueline	17.05.16
3 Daniel	15.05.16


```

SELECT first_name, hire_date
FROM employee
WHERE hire_date BETWEEN '01.05.2016' AND '31.05.2016';
;
```

Query Result | All Rows Fetched: 3 in 0,005 seconds

FIRST_NAME	HIRE_DATE
1 John	25.05.16
2 Jacqueline	17.05.16
3 Daniel	15.05.16

ALL keywords

ALL — bu barcha shartni qanoatlantiradigan qiymatlar chiqarilsin.

```

SELECT first_name, salary
FROM employee
WHERE salary > 30000
AND salary > 40000
AND salary > 50000;
;
```

Query Result | Fetched 50 rows in 0,007 seconds

FIRST_NAME	SAL...
1 Patrick	51000
2 Sean	51000
3 Donna	51000
4 Martha	52000


```

SELECT first_name, salary
FROM employee
WHERE salary > ALL(30000, 40000, 50000)
;
```

Query Result | Fetched 50 rows in 0,007 seconds

FIRST_NAME	SAL...
1 Patrick	51000
2 Sean	51000
3 Donna	51000
4 Martha	52000


```

SELECT first_name, salary
FROM employee
WHERE salary >= ALL(30000, 40000, 50000)
;
```

Query Result | Fetched 50 rows in 0,004 seconds

FIRST_NAME	SAL...
1 Daniel	50000
2 Justin	50000
3 Patrick	51000
4 Sean	51000

The image shows two side-by-side SQL query windows. Both queries select first names and salaries from the employee table where the salary is less than or equal to 30,000, 40,000, or 50,000.

```

    SELECT first_name, salary
    FROM employee
    WHERE salary <= ALL(30000, 40000, 50000)
;

```

Query Result:

FIRST_NAME	SAL...
1 Kenneth	29000
2 Phyllis	29000
3 Wanda	28000


```

    SELECT first_name, salary
    FROM employee
    WHERE salary < ALL(30000, 40000, 50000)
;

```

Query Result:

FIRST_NAME	SAL...
1 Joseph	30000
2 Kenneth	29000
3 Phyllis	29000

The image shows two side-by-side SQL query windows. The first query selects employees whose salary is not equal to 30,000, 40,000, or 50,000. The second query selects employees whose salary is equal to 30,000, 40,000, or 50,000.

```

    SELECT first_name, salary
    FROM employee
    WHERE salary <> ALL(30000, 40000, 50000)
;

```

Query Result:

FIRST_NAME	SALARY
1 Michelle	48000
2 Cheryl	79000
3 Carolyn	47000


```

    SELECT first_name, salary
    FROM employee
    WHERE salary = ALL(30000, 40000, 50000)
;

```

Query Result:

FIRST_NAME	SALARY
------------	--------

ANY keywords

ANY – bu birorta shartni qanoatlantiradigan qiymatlar chiqarilsin.

The image shows two side-by-side SQL query windows. Both queries select first names and salaries from the employee table where the salary is greater than 30,000, 40,000, or 50,000.

```

    SELECT first_name, salary
    FROM employee
    WHERE salary > 30000
    OR salary > 40000
    OR salary > 50000
;

```

Query Result:

FIRST_NAME	SAL...
1 Amy	31000
2 Susan	31000
3 Theresa	32000


```

    SELECT first_name, salary
    FROM employee
    WHERE salary > ANY(30000, 40000, 50000)
;

```

Query Result:

FIRST_NAME	SAL...
1 Amy	31000
2 Susan	31000
3 Theresa	32000

```

SELECT first_name, salary
FROM employee
WHERE salary = ANY(30000, 40000, 50000)
;

Query Result x
SQL | All Rows Fetched: 6 in 0,005 seconds


|   | FIRST_NAME | SALARY |
|---|------------|--------|
| 1 | Roy        | 40000  |
| 2 | Justin     | 50000  |
| 3 | Christine  | 40000  |
| 4 | Joseph     | 30000  |
| 5 | Kelly      | 40000  |
| 6 | Daniel     | 50000  |


```

SORTING DATA

ORDER BY — bu tartiblash.

ASC – O'sish tartibi, **DESC** – kamayish tartibi.

ORDER BY (1ta ustun uchun)

```

SELECT employee_id, last_name
FROM employee
ORDER BY last_name ASC
;

Query Result x
SQL | Fetched 50 rows in 0,004 s


| EMPLOYEE_ID | LAST_NAME  |
|-------------|------------|
| 1           | 20 Adams   |
| 2           | 71 Allen   |
| 3           | 40 Alvarez |


```

```

SELECT employee_id, last_name
FROM employee
ORDER BY last_name DESC
;

Query Result x
SQL | Fetched 50 rows in 0,004 s


| EMPLOYEE_ID | LAST_NAME   |
|-------------|-------------|
| 1           | 102 Woods   |
| 2           | 94 Williams |
| 3           | 95 Williams |


```

```

SELECT last_name, salary
FROM employee
ORDER BY salary ASC
;

Query Result x
SQL | Fetched 50 rows in 0,004 s


| LAST_NAME  | SALARY |
|------------|--------|
| 1 Elliott  | 21000  |
| 2 Cook     | 21000  |
| 3 Sullivan | 21000  |


```

```

SELECT last_name, salary
FROM employee
ORDER BY salary DESC
;

Query Result x
SQL | Fetched 50 rows in 0,004 s


| LAST_NAME   | SALARY |
|-------------|--------|
| 1 Fisher    | (null) |
| 2 Rice      | (null) |
| 3 Henderson | 120000 |


```

ORDER BY (bir nechta ustun uchun)

Last_name bo'yicha kamayuvchi, First_name bo'yicha o'suvchi holda tartiblash.

First_name bo'yicha tartiblaydi, qachonki, Last_name lari bir xil bo'lganlarini.

The screenshot shows a MySQL query editor window. The SQL query is:

```
SELECT employee_id, last_name, first_name
FROM employee
ORDER BY last_name DESC, first_name ASC
;
```

The results are displayed in a table titled "Query Result". The table has three columns: EMPLOYEE_ID, LAST_NAME, and FIRST_NAME. The data is as follows:

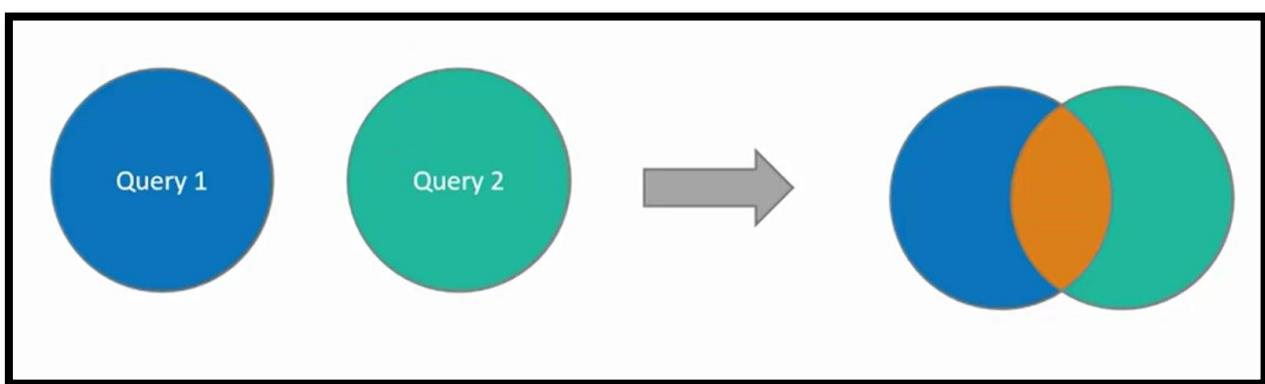
EMPLOYEE_ID	LAST_NAME	FIRST_NAME
1	102 Woods	Ralph
2	61 Williams	Karen
3	94 Williams	Lisa
4	95 Williams	Stephanie

Below the table, it says "Fetched 50 rows in 0,004 seconds".

SET OPERATORS

UNION

UNION



UNION — jadvallarni birlashtirish uchun ishlataladi.

1-holat. O'xshashlari birlashtirib chiqariladi. (207ta qiymat)

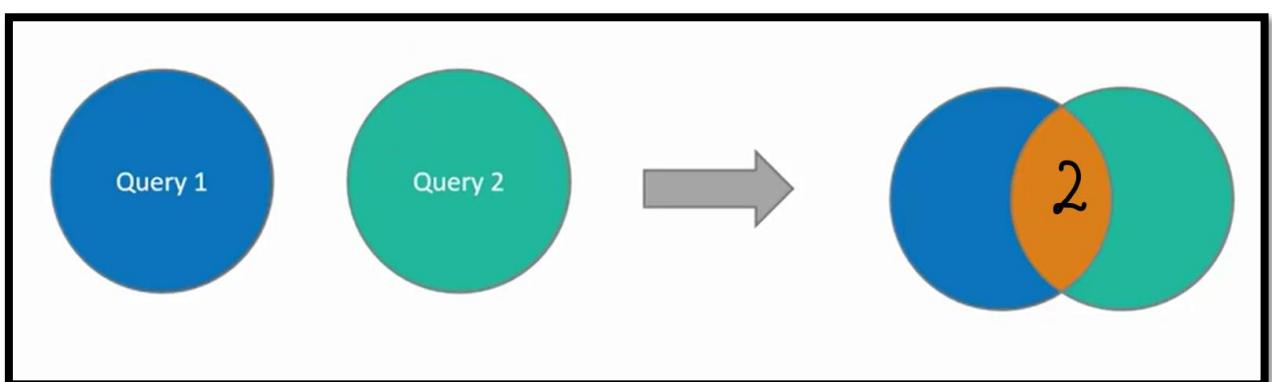
2-holat. Har biri alohida o'z unvoni bilan chiqadi. (210ta qiymat)

3-holat. **ORDER BY** bilan tartiblab chiqardik va o'xshashlarini topdik.

4-holat. **ORDER BY** 1, 2 holatida chiqarildi. Oldin mijozlar keyin xodimlar *first_name* bo'yicha tartiblanib chiqarildi. Buni hali batafsil keyinroq o'rganamiz.

<pre>SELECT first_name, last_name FROM employee UNION SELECT first_name, last_name FROM customer;</pre>	<pre>SELECT 'Employee', first_name, last_name FROM employee UNION SELECT 'Customer', first_name, last_name FROM customer;</pre>																																
<p>Query Result x</p> <p>All Rows Fetched: 207 in 0,027 seconds</p> <table border="1"><thead><tr><th></th><th>FIRST_NAME</th><th>LAST_NAME</th></tr></thead><tbody><tr><td>1</td><td>Michelle</td><td>Foster</td></tr><tr><td>2</td><td>Cheryl</td><td>Turner</td></tr><tr><td>3</td><td>Carolyn</td><td>Hudson</td></tr></tbody></table>		FIRST_NAME	LAST_NAME	1	Michelle	Foster	2	Cheryl	Turner	3	Carolyn	Hudson	<p>Query Result x</p> <p>All Rows Fetched: 210 in 0,027 seconds</p> <table border="1"><thead><tr><th></th><th>'EMPLOYEE'</th><th>FIRST_NAME</th><th>LAST_NAME</th></tr></thead><tbody><tr><td>199</td><td>Employee</td><td>Lawrence</td><td>Henderson</td></tr><tr><td>200</td><td>Employee</td><td>Charles</td><td>Stone</td></tr><tr><td>201</td><td>Customer</td><td>Teresa</td><td>Hudson</td></tr></tbody></table>		'EMPLOYEE'	FIRST_NAME	LAST_NAME	199	Employee	Lawrence	Henderson	200	Employee	Charles	Stone	201	Customer	Teresa	Hudson				
	FIRST_NAME	LAST_NAME																															
1	Michelle	Foster																															
2	Cheryl	Turner																															
3	Carolyn	Hudson																															
	'EMPLOYEE'	FIRST_NAME	LAST_NAME																														
199	Employee	Lawrence	Henderson																														
200	Employee	Charles	Stone																														
201	Customer	Teresa	Hudson																														
<pre>SELECT 'Employee', first_name, last_name FROM employee UNION SELECT 'Customer', first_name, last_name FROM customer ORDER BY last_name;</pre>	<pre>SELECT 'Employee', first_name, last_name FROM employee UNION SELECT 'Customer', first_name, last_name FROM customer ORDER BY 1,2;</pre>																																
<p>Query Result x</p> <p>All Rows Fetched: 210 in 0,027 seconds</p> <table border="1"><thead><tr><th></th><th>'EMPLOYEE'</th><th>FIRST_NAME</th><th>LAST_NAME</th></tr></thead><tbody><tr><td>130</td><td>Employee</td><td>Bruce</td><td>Mitchell</td></tr><tr><td>131</td><td>Employee</td><td>Ralph</td><td>Montgomery</td></tr><tr><td>132</td><td>Customer</td><td>Ralph</td><td>Montgomery</td></tr></tbody></table>		'EMPLOYEE'	FIRST_NAME	LAST_NAME	130	Employee	Bruce	Mitchell	131	Employee	Ralph	Montgomery	132	Customer	Ralph	Montgomery	<p>Query Result x</p> <p>Fetched 50 rows in 0,005 seconds</p> <table border="1"><thead><tr><th></th><th>'EMPLOYEE'</th><th>FIRST_NAME</th><th>LAST_NAME</th></tr></thead><tbody><tr><td>1</td><td>Customer</td><td>Alice</td><td>Perry</td></tr><tr><td>2</td><td>Customer</td><td>Dorothy</td><td>Armstrong</td></tr><tr><td>3</td><td>Customer</td><td>Fred</td><td>Wallace</td></tr></tbody></table>		'EMPLOYEE'	FIRST_NAME	LAST_NAME	1	Customer	Alice	Perry	2	Customer	Dorothy	Armstrong	3	Customer	Fred	Wallace
	'EMPLOYEE'	FIRST_NAME	LAST_NAME																														
130	Employee	Bruce	Mitchell																														
131	Employee	Ralph	Montgomery																														
132	Customer	Ralph	Montgomery																														
	'EMPLOYEE'	FIRST_NAME	LAST_NAME																														
1	Customer	Alice	Perry																														
2	Customer	Dorothy	Armstrong																														
3	Customer	Fred	Wallace																														

UNION ALL



UNION ALL — barchasini qo'shib ketma-ket chiqarishdir.

```

SELECT first_name, last_name
FROM employee
UNION ALL
SELECT first_name, last_name
FROM customer
ORDER BY 1,2;

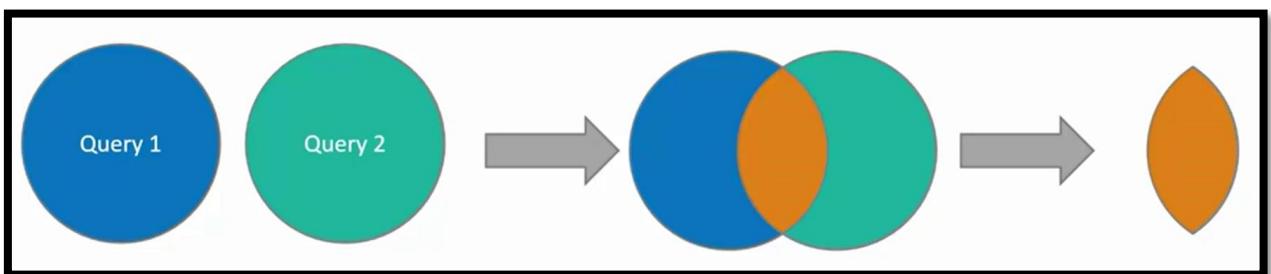
```

Query Result | SQL | All Rows Fetched: 210 in

	FIRST_NAME	LAST_NAME
1	Adam	Stephens
2	Alice	Perry
3	Alice	Perry

INTERSECT

Intersect — bu kesishmani chiqarish. Ya’ni ikki jadvalda ham bir xil takrorlangan qiymatlarni chiqarish.



```

SELECT first_name, last_name
FROM employee
INTERSECT
SELECT first_name, last_name
FROM customer;

```

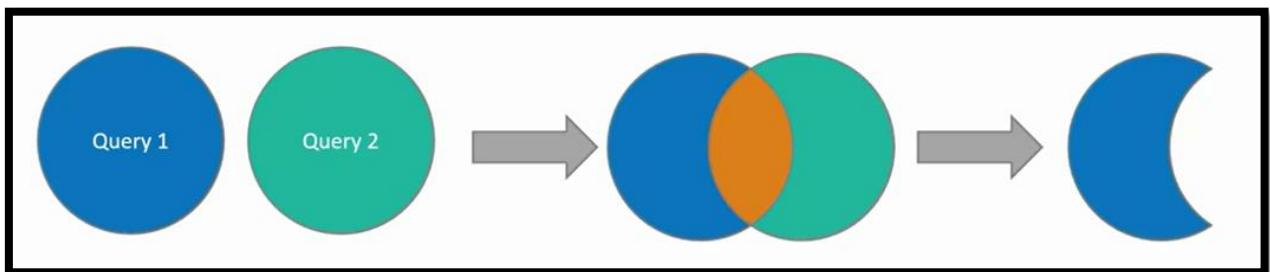
Query Result | SQL | All Rows Fetched: 3 in 0.000 sec

	FIRST_NAME	LAST_NAME
1	Alice	Perry
2	Ralph	Montgomery
3	Fred	Wallace

- Query A has records 1, 2, 3, 4
- Query B has records 2, 4, 6, 8, 9
- UNION: 1, 2, 3, 4, 6, 8, 9
- UNION ALL: 1, 2, 2, 3, 4, 4, 6, 8, 9
- INTERSECT: 2, 4

MINUS

MINUS — bu 1-to'plamdan kesishmani olib tashlab chiqarish.



1-holat. Barcha hodimlar. (200 ta)

2-holat. Hodimlar va haridorlar kesishmasi. (3 ta)

3-holat. Hodimlardan haridorlar ayirmasi. ($200 - 3 = 197$ ta)

```
SELECT first_name, last_name
FROM employee;
SELECT first_name, last_name
FROM customer;
```

```
SELECT first_name, last_name
FROM employee
INTERSECT
SELECT first_name, last_name
FROM customer;
```

```
SELECT first_name, last_name
FROM employee
MINUS
SELECT first_name, last_name
FROM customer;
```

- Query A has records 1, 2, 3, 4
- Query B has records 2, 4, 6, 8, 9
- UNION: 1, 2, 3, 4, 6, 8, 9
- UNION ALL: 1, 2, 2, 3, 4, 4, 6, 8, 9
- INTERSECT: 2, 4
- MINUS: 1, 3

MORE SETS

SET OPERATOR lar kombinatsiyasi.

1-holat. UNION ([28;40] U [20;30] => [20;40])

2-holat. UNION + INTERSECT ([20;40] () [30;55] => [30;40])

3-holat. UNION + INTERSECT + MINUS (↓)

$$([30;40] - [31;33] => [30] \cup [34;40])$$

4-holat. INTERSECT + UNION + MINUS

```

SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 28000 AND 40000
UNION
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 20000 AND 30000;

```

Query Result

FIRST_NAME	LAST_NAME	SAL...
1 Jessica	Sullivan	21000
2 Jessica	Elliott	21000
3 Dorothy	Cook	21000

```

SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 28000 AND 40000
UNION
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 20000 AND 30000
INTERSECT
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 30000 AND 55000
;
```

Query Result

FIRST_NAME	LAST_NAME	SAL...
1 Joseph	Davis	30000
2 Amy	Wheeler	31000
3 Susan	Henderson	31000

```

SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 28000 AND 40000
UNION
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 20000 AND 30000
INTERSECT
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 30000 AND 55000
MINUS
SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 31000 AND 33000
;
```

Query Result

FIRST_NAME	LAST_NAME	SAL...
1 Joseph	Davis	30000
2 Ann	Bowman	34000
3 Ralph	Anderson	34000

2

```

SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 28000 AND 40000
UNION
2 (SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 20000 AND 30000
INTERSECT
1 SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 30000 AND 55000)
MINUS
3 SELECT first_name, last_name, salary
FROM employee
WHERE salary BETWEEN 31000 AND 33000
;
```

Query Result

FIRST_NAME	LAST_NAME	SAL...
1 Wanda	Stewart	28000
2 Phyllis	Larson	29000
3 Kenneth	Little	29000

AGGREGATE FUNCTIONS AND GROUPING

COUNT()

COUNT() — qiymatlar sonini chiqaradi.

COUNT(*) — satrlar sonini chiqaradi.

1-holat. Xodimlar ma'lumotlarini chiqaradi.

2-holat. Barbara ismli xodimlarni ma'lumotlarini chiqaradi.

3-holat. Barbara ismli xodimlar sonini chiqaradi.

The image shows three separate MySQL Workbench windows demonstrating the COUNT function.

- Left Window:** Shows the query `SELECT COUNT(*) FROM employee;`. The result table has one row with `COUNT(*)` value 200.
- Middle Window:** Shows the query `SELECT * FROM employee WHERE first_name = 'Barbara';`. The result table lists four rows of employee data where the first name is 'Barbara'.
- Right Window:** Shows the query `SELECT COUNT(*) FROM employee WHERE first_name = 'Barbara';`. The result table has one row with `COUNT(*)` value 4.

4-holat. Ish haqi oladiganlar soni (198 ta qiymat)

5-holat. Ish haqi olmaydiganlar ma'lumotlari (2 ta **NULL** katak)

The image shows two separate MySQL Workbench windows demonstrating COUNT with different conditions.

- Left Window:** Shows the query `SELECT COUNT(salary) FROM employee;`. The result table has one row with `COUNT(SALARY)` value 198.
- Right Window:** Shows the query `SELECT * FROM employee WHERE salary IS NULL;`. The result table lists two rows where the salary is null, corresponding to employees Anthony Rice and Jennifer Fish.

6-holat. Faqrli ismlar chiqarilsin.

7-holat. Farqli ismlar soni chiqarilsin.

8-holat. Qatorlar sonini hisoblandi, va hisobni har xilligi tekshirildi

The image displays three separate SQL query windows side-by-side. Each window has a 'Query Result' tab at the top.

- Left Window:** Shows the query `SELECT DISTINCT last_name FROM employee;`. The result table has one column named 'LAST_NAME' with three rows: 1 Foster, 2 Turner, and 3 Hudson.
- Middle Window:** Shows the query `SELECT COUNT(DISTINCT last_name) FROM employee;`. The result table has one column named 'COUNT(DISTINCT LAST_NAME)' with one row containing the value 140.
- Right Window:** Shows the query `SELECT DISTINCT COUNT(last_name) FROM employee;`. The result table has one column named 'COUNT(LAST_NAME)' with one row containing the value 200.

Count data within GROUP BY

GROUP BY — bu guruhlash

Shart. Familiyalar chiqarilsin, ularning takrorlanish sonlari chiqarilsin.

1-holat. Ishlamaydi. Chunki *last_name*da qator ko'p, countda 1ta.

2-holat. Ishlaydi.

The image shows two adjacent SQL query windows.

- Left Window:** Contains the query `SELECT last_name, COUNT(*) FROM employee;`. Below the query, an error message is displayed: ORA-00937: групповая функция не является одноточечной. 00937. 00000 - "not a single-group group function".
- Right Window:** Contains the query `SELECT last_name, COUNT(*) FROM employee GROUP BY last_name;`. The result table has two columns: 'LAST_NAME' and 'COUNT(*)'. It shows three rows: Foster (count 2), Turner (count 1), and Hudson (count 2).

GROUP BY with ORDER BY

1-holat. Familiyalar takrorlanishlar soni kamayuvchi tartibi.

2-holat. *Last_name* va *department_id*lari aynan bir xil bo'lganlar sonini kamayuvchi holatda tartiblash. [GROUP BYda SELECTda yozilgan ustunlar hammasi yozilishi shart.]

```

SELECT last_name, COUNT(*)
FROM employee
GROUP BY last_name
ORDER BY COUNT(*) DESC;

```

```

SELECT last_name, department_id, COUNT(*)
FROM employee
GROUP BY last_name, department_id
ORDER BY COUNT(*) DESC;

```

LAST_NAME	COUNT(*)
1 Diaz	4
2 Berry	3
3 Collins	3

LAST_NAME	DEPARTMENT_ID	COUNT(*)
1 Jones		7
2 Gonzalez		5
3 Rogers		7

GROUP BY with HAVING

Shart. Familiyalarning takrorlanish sonlari 1dan ortig'i chiqarilsin.

1-holat. Ishlamaydi. Chunki WHEREda Group Function ishlatalmaydi.

2-holat. Ishlaydi. HAVING ishlatalindi.

3-holat. Familiyalari va oyliklari bir xillar soni 2tadan oshig'i chiqarish.

```

SELECT last_name, COUNT(*)
FROM employee
WHERE COUNT(*) > 1
GROUP BY last_name;

```

```

ORA-00934: групповая функция здесь не разрешена
00934. 00000 - "group function is not allowed here"
*Cause:
*Action:
Error at Line: 353 Column: 7

```

```

SELECT last_name, COUNT(*)
FROM employee
GROUP BY last_name
HAVING COUNT(*) > 1;

```

LAST_NAME	COUNT(*)
1 Foster	2
2 Hudson	2
3 Berry	3

```

SELECT last_name, salary, COUNT(*)
FROM employee
GROUP BY last_name, salary
HAVING COUNT(*) > 1;

```

LAST_NAME	SALARY	COUNT(*)
1 Richardson	80000	2
2 Howell	102000	2

SUM function

SUM — yig'indi.

1-holat. Oyliklar yig'indisi.

2-holat. Department_id'lari bir xil xodimlarning oyliklari yig'indisi

```

SELECT SUM(salary)
FROM employee;

```

	DEPARTMENT_ID	SUM(SALARY)
1	8	1560000
2	3	2233000
3	7	1587000


```

SELECT department_id, SUM(salary)
FROM employee
GROUP BY department_id;

```

MAX() and MIN() function

1-holat. Maksimal oylik narxini chiqarish.

2-holat. Minimal oylik narxini chiqarish.

3-holat. Maksimal va Minimal oylik narxini chiqarish.

```

SELECT MAX(salary)
FROM employee;

```

	MAX(SALARY)
1	120000


```

SELECT MIN(salary)
FROM employee;

```

	MIN(SALARY)
1	21000

```

SELECT MAX(salary), MIN(salary)
FROM employee;

```

	MAX(SALARY)	MIN(SALARY)
1	120000	21000

4-holat. *Department_id*lari o'zaro bir xil bo'lganlarning eng katta, eng kichik va yig'indi oylik narxlari

```
SELECT department_id, MAX(salary), MIN(salary), SUM(salary)
FROM employee
GROUP BY department_id;
```

Query Result | All Rows Fetched: 9 in 0,007 seconds

DEPARTMENT_ID	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)	
1	8	120000	23000	1560000
2	3	120000	25000	2233000
3	7	114000	21000	1587000

5-holat. MAX() va MIN() sanalarda ham ishlaydi.

```
SELECT MIN(hire_date), MAX(hire_date)
FROM employee;
```

Query Result | All Rows Fetched: 1 in 0,002 seconds

MIN(HIRE_DATE)	MAX(HIRE_DATE)
03.01.10	24.01.17

6-holat. MAX() va MIN() harflarda ham ishlaydi.

```
SELECT MIN(first_name), MAX(last_name)
FROM employee;
```

Query Result | All Rows Fetched: 1 in 0,004 seconds

MIN(FIRST_NAME)	MAX(LAST_NAME)
Adam	Woods

AVG function

1-holat. *Department_id*lari o'zaro bir xil bo'lganlarning o'rtacha oylik narxlari

```

SELECT department_id, AVG(salary)
FROM employee
GROUP BY department_id;

```

Query Result | All Rows Fetched: 9 in 0,006 seconds

DEPARTMENT_ID	AVG(SALARY)	
1	8	62400
2	3	69781,25
3	7	56678,5714285714285714285714285714285714

2-holat. Nomuvofiq datatip. Kutilyapti NUMBER, kelayotgani DATE

```

SELECT AVG(hire_date)
FROM employee;

```

Query Result | Executing:SELECT AVG(hire_date)FROM employee in 0 sec

ORA-00932: несовместимые типы данных: ожидается NUMBER, получено DATE
00932. 00000 - "inconsistent datatypes: expected %s got %s"

JOINS

TABLE ALIASES

TAXALLUS — uzun nomlarni yozish uchun ketadigan vaqtni tejab beradi.

```

SELECT last_name, e.salary
FROM employee e
WHERE e.salary < 30000;

```

Query Result | All Rows Fetched: 22 in 0,006 seconds

LAST_NAME	SAL...
1 Little	29000
2 Larson	29000
3 Stewart	28000

```

SELECT last_name, e.salary AS annual_salary
FROM employee e;

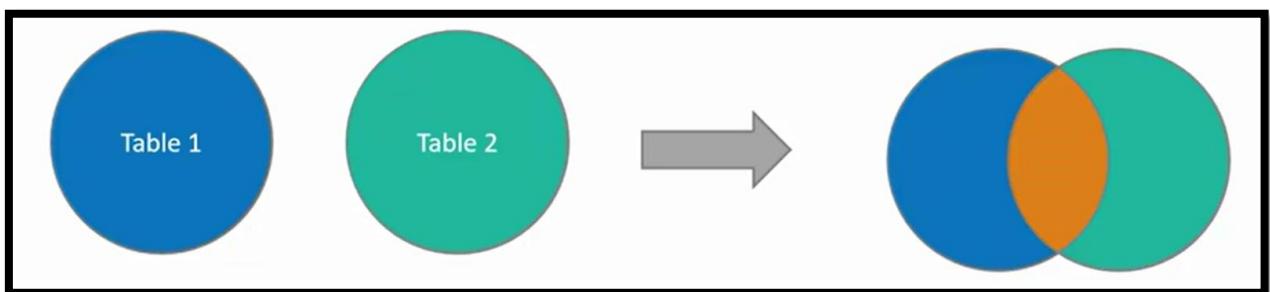
```

Query Result | Fetched 50 rows in 0,013 seconds

LAST_NAME	ANNUAL_SALARY
1 Foster	48000
2 Turner	79000
3 Hudson	47000

INNER JOIN

JOIN — 2ta jadvalni qo'shib chiqarishda foydalaniladi. 1 ta so'rovda bir nechta jadval ma'lumotlarini chiqarishda foydalaniladi.



```
SELECT employee_id,
       first_name,
       last_name,
       department_name
  FROM employee
 JOIN department ON employee.department_id = department.department_id;
```

Query Result | All Rows Fetched: 197 in 0,048 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	
1	1	Michelle	Foster	Legal	
2	2	Cheryl	Turner	Customer Support	
3	3	Carolyn	Hudson	Finance	

Pastda xatolik yuz berdi. Sababi *department_id* ustuni qaysi jadvalga tegishli ekanligi aniqlanmadi.

```
SELECT employee_id,
       first_name,
       last_name,
       department_name,
       department_id -- department.department_id (to'g'ri holati)
  FROM employee
 JOIN department ON employee.department_id = department.department_id;
```

Query Result x

SQL | Executing:SELECT employee_id, first_name, last_name, department_name,depart

ORA-00918: столбец определен неоднозначно
00918. 00000 - "column ambiguously defined"

Taxalluslar bilan nomlarni qisqartirdik.

```
SELECT e.employee_id,
       e.first_name,
       e.last_name,
       d.department_name,
       d.department_id
  FROM employee e
 JOIN department d ON e.department_id = d.department_id;
```

Query Result x

SQL | Fetched 50 rows in 0,017 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	DEPARTMENT_ID
1	1	Michelle	Foster	Legal	8
2	2	Cheryl	Turner	Customer Support	3
3	3	Carolyn	Hudson	Finance	7

Mana shu so'rovga shart kiritamiz.

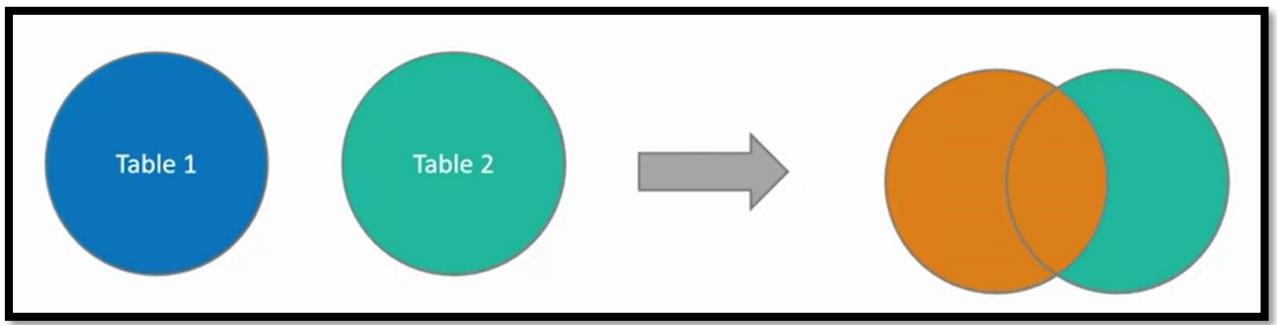
```
SELECT e.employee_id,
       e.first_name,
       e.last_name,
       d.department_name,
       d.department_id,
       e.salary
  FROM employee e
 JOIN department d ON e.department_id = d.department_id
 WHERE e.salary > 60000;
```

Query Result x

SQL | Fetched 50 rows in 0,007 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	DEPARTMENT_ID	SAL...
1	117	Anna	Green	Customer Support	3	61000
2	36	Teresa	Graham	Marketing	6	61000
3	13	Stephen	Hudson	Hardware Development	4	63000

LEFT JOIN



INNER JOIN holatida faqat kesishmalar olinadi. (50ta qiymat)

Query:

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    co.order_date
FROM customer c
INNER JOIN customer_order co ON c.customer_id = co.customer_id;
```

Query Result:

CUSTOMER_ID	FIRST_NAME	LAST_NAME	ORDER_DATE
1	3 Lois	Lawson	18.12.16
2	5 Ralph	Montgomery	01.06.15
3	6 Dorothy	Armstrong	26.09.16

LEFT JOIN holatida chapdagи barcha qiymatlar olinadi. Kesishmagan joyi *NULL* bilan to'ldiriladi. (52ta qiymat)

Query:

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    co.order_date
FROM customer c
LEFT JOIN customer_order co ON c.customer_id = co.customer_id;
```

Query Result:

CUSTOMER_ID	FIRST_NAME	LAST_NAME	ORDER_D...
1	1 Teresa	Hudson	(null)
2	4 Alice	Perry	(null)
3	2 Fred	Montgomery	23.01.17

Department_id = 9 bo'lgan qismda hodimlar yo'qligi aniqlandi.

```
SELECT
d.department_id,
d.department_name,
e.employee_id,
e.first_name,
e.last_name
FROM department d
LEFT JOIN employee e ON d.department_id = e.department_id
ORDER BY department_id;
```

Query Result x
SQL | Fetched 50 rows in 0,007 seconds

DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEE_ID	FIRST_NAME	LAST_N...
1	9 Maintenance	(null)	(null)	(null)
2	4 Hardware Deve...	102	Ralph	Woods
3	3 Customer Support	61	Karen	Williams

Tekshirib ko'ramiz.

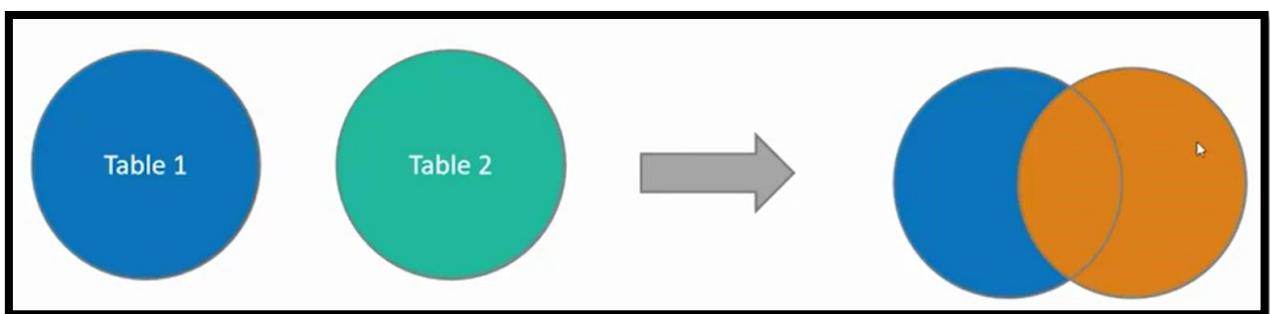
```
SELECT first_name, last_name
FROM employee
WHERE department_id = 9;
```

Query Result x
SQL | All Rows Fetched: 0 in 0

FIRST_NA...	LAST_NAME
-------------	-----------

RIGHT JOIN

RIGHT JOIN => *LEFT JOIN*ning huddi o'zginasi. Farqi bu o'ng tarafni oladi xolos.



2ta sinonim holat. Biri *LEFT JOIN* biri *RIGHT JOIN*

```
SELECT
    d.department_id,
    d.department_name,
    e.employee_id,
    e.first_name,
    e.last_name
FROM department d
LEFT JOIN employee e ON d.department_id = e.department_id
ORDER BY department_id;
```

Query Result | Fetched 50 rows in 0,007 seconds

	DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEE_ID	FIRST_NAME	LAST_N...
1	9 Maintenance		(null) (null)	(null)	
2	4 Hardware Deve...		102 Ralph	Woods	
3	3 Customer Support		61 Karen	Williams	

```
SELECT
    d.department_id,
    d.department_name,
    e.employee_id,
    e.first_name,
    e.last_name
FROM employee e
RIGHT JOIN department d ON e.department_id = d.department_id
ORDER BY department_id;
```

Query Result | All Rows Fetched: 198 in 0,022 seconds

	DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEE_ID	FIRST_NAME	LAST_N...
1	9 Maintenance		(null) (null)	(null)	
2	4 Hardware Deve...		102 Ralph	Woods	
3	3 Customer Support		61 Karen	Williams	

Employee va Department o'rnnini almashtiramiz. Departmentsiz (hech bir bo'limga ega bo'limgan) hodimlar NULL deb chiqarildi.

```

SELECT
    d.department_id,
    d.department_name,
    e.employee_id,
    e.first_name,
    e.last_name
FROM department d
RIGHT JOIN employee e ON d.department_id = e.department_id
ORDER BY department_id;

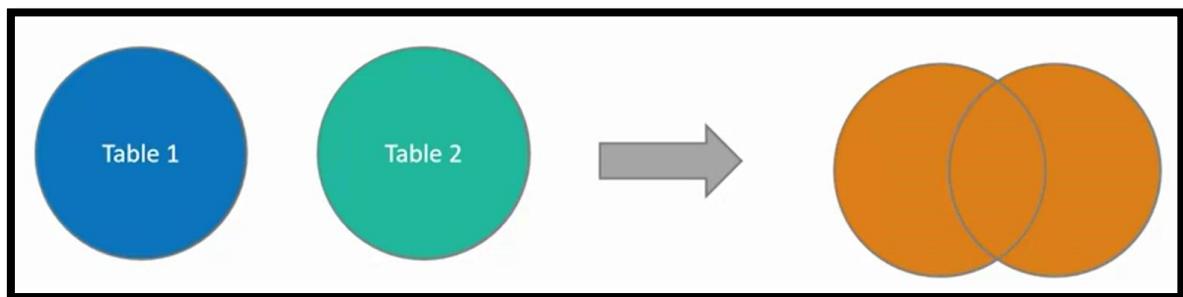
```

Query Result | Fetched 50 rows in 0,007 seconds

	DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEE_ID	FIRST_NAME	LAST_NAME
1	(null) (null)		84	Jennifer	Long
2	(null) (null)		140	Rebecca	Chapman
3	(null) (null)		114	James	Phillips
4	8	Legal	77	David	Thompson

FULL JOIN

FULL JOIN — barcha jadvallarni qamrab oladi.



```

SELECT
    d.department_id,
    d.department_name,
    e.employee_id,
    e.first_name,
    e.last_name
FROM department d
FULL JOIN employee e ON d.department_id = e.department_id
ORDER BY d.department_id, e.employee_id;

```

Query Result | All Rows Fetched: 201 in 0,018 seconds

	DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEE_ID	FIRST_NAME	LAST_NAME
1	(null) (null)		84	Jennifer	Long
2	(null) (null)		114	James	Phillips
3	(null) (null)		140	Rebecca	Chapman
4	9	Maintenance	(null) (null)	(null)	
5	8	Legal	1	Michelle	Foster

Bunda 201ta qiymat aniqlandi sababi. *Employeeda* 200ta qiymat bor va 9-bo'limda hodim yo'qligi ko'rsatilgan.

NATURAL JOIN

NATURAL JOIN. Nomiari bir xil bo'lgan ustunlar qo'shilishida va 1-jadvaldagi asosiy kalit 2-jadvaldagi tashqi kalit bilan bil xil bo'lganda foydalaniladi.

Lekin foydalanmaslik tavsiya etiladi.

The screenshot shows a MySQL Workbench interface. In the SQL editor, a query is written:

```
SELECT
    e.employee_id,
    e.first_name,
    e.last_name,
    department_id,
    d.department_name
FROM employee e
NATURAL JOIN department d;
```

The 'department_id' column is underlined in red. In the 'Query Result' tab, the output is:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	1	Michelle	Foster	8	Legal
2	2	Cheryl	Turner	3	Customer Support
3	3	Carolyn	Hudson	7	Finance

Ustun nomlari bir xil bo'lgani uchun ishladi.

Yana bir misol:

The screenshot shows a MySQL Workbench interface. In the SQL editor, a query is written:

```
SELECT product_id,
       product_name,
       department_id,
       department_name
FROM product
NATURAL JOIN department;
```

The 'department_id' column is underlined in red. In the 'Query Result' tab, the output is:

	PRODUCT_ID	PRODUCT_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	1	Monitor	4	Hardware Development
2	2	Desk	4	Hardware Development
3	3	Chair	4	Hardware Development

CARTESIAN or CROSS JOIN

2ta jadval berilgan satrlari (200 va 9) CROSS JOIN ishlatilsa, natijaviy satrlar soni $200 \times 9 = 1800$ ta bo'ladi.

Bu yerda 200ta ism executive bo'lib chiqdi va 1-dan boshlab yana 200 tasi sales bo'lib chiqadi va h.z.

The screenshot shows a MySQL Workbench interface. In the SQL editor, the following query is written:

```
SELECT employee_id, first_name, last_name,
       department_name
  FROM employee, department;
```

In the Query Result tab, the output is displayed as a table:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME
199	199	Lawrence	Henderson	Executive
200	200	Charles	Stone	Executive
201	1	Michelle	Foster	Sales

The total number of rows fetched is 1800, as indicated in the status bar.

SELF JOIN

SELF JOIN — O'z-o'ziga qo'shilish.

Bunda 1ta jadvalda bir ustun boshqa bir ustunga bog'langan bo'ladi.

Misol uchun hodimlar jadvalida 1 hodimning menejeri bor, u menejer ham asli bir hodim, u menejerning ham menejeri bo'lishi mumkin. Ya'ni quyidagicha:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	MANAGER_ID
1	Michelle	Foster	162
162	Kimberly	Mendoza	191
191	Randy	Spencer	193

Employee jadvalidan misol ko'ramiz.

```

SELECT employee_id,
       first_name,
       last_name,
       manager_id
  FROM employee;

```

Query Result

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	MANAGER_ID
1	Michelle	Foster	162
2	Cheryl	Turner	99
3	Carolyn	Hudson	199

SELF JOINni ishlatdik. Manager_id = employee_id

```

SELECT emp.employee_id,
       emp.first_name,
       emp.last_name,
       emp.manager_id,
       mgr.first_name,
       mgr.last_name
  FROM employee emp
 LEFT JOIN employee mgr ON emp.manager_id = mgr.employee_id;

```

Query Result

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	MANAGER_ID	FIRST_NAME_1	LAST_NAME_1
1	Doris	Powell	(null)	(null)	(null)
2	Teresa	Bell	200	Charles	Stone
3	Pamela	Collins	199	Lawrence	Henderson
4	Carolyn	Hudson	199	Lawrence	Henderson

NULL qiymatli qator mavjudligining sababi *id=5* bo'lgan hodimning menejeri mavjud emas. Uning o'zi eng katta menejer yoki shunchaki ish hamkori.

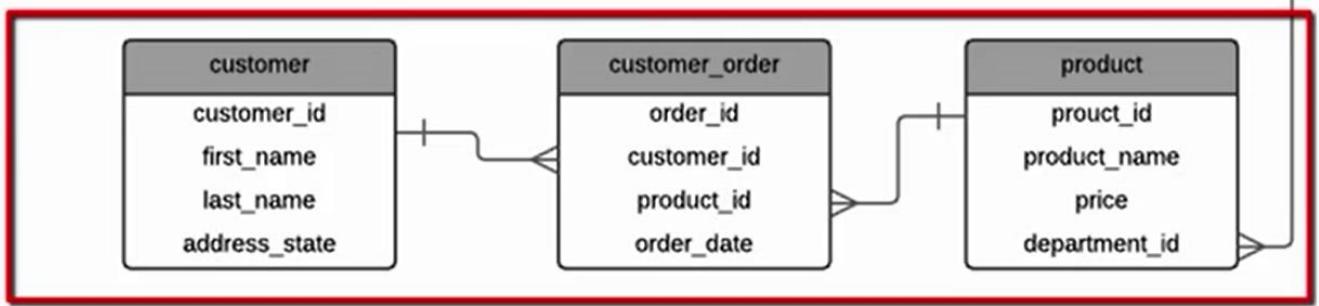
JOINING MANY TABLES

Bir nechta ustunlar uchun JOINlarni qo'llaymiz.

- SELECT ... FROM table1
JOIN table2 ON table1.col = table2.col
JOIN table3 ON table2.col = table3.col

- Combination of INNER, LEFT, RIGHT, etc...

Bizning bazamiz arxitekturasining bir qismi:



Bu yerda haridor va haridlarni bir-biriga bog'liq holda chiqardik:

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.address_state,
    co.order_id,
    co.order_date
FROM customer c
JOIN customer_order co ON c.customer_id = co.customer_id;
```

Query Result | Fetched 50 rows in 0,006 seconds

	CUSTOMER_ID	FIRST_NAME	LAST_NAME	ADDRESS_STATE	ORDER_ID	ORDER_DATE
1	3	Lois	Lawson	OR	1	18.12.16
2	5	Ralph	Montgomery	TX	2	01.06.15
3	6	Dorothy	Armstrong	OR	3	26.09.16

Bu yerda esa Customer Customer_order va Product jadvallarini bog'liq holda chiqardik:

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.address_state,
    co.order_id,
    co.order_date,
    p.product_name,
    p.price
FROM customer c
JOIN customer_order co ON c.customer_id = co.customer_id
JOIN product p ON co.product_id = p.product_id;
```

Query Result | Fetched 50 rows in 0,004 seconds

	CUSTOMER_ID	FIRST_NAME	LAST_NAME	ADDRESS_STATE	ORDER_ID	ORDER_DATE	PRODUCT_NAME	PRICE
1	2	Fred	Montgomery	CA	9	23.01.17	Photo Editing Pro	250
2	2	Fred	Montgomery	CA	11	09.06.16	Desk	110,9
3	2	Fred	Montgomery	CA	33	28.10.16	Monitor	149,95

Bu yerda Customer_id bo'yicha tartiblandi va bundan har bir haridor haridlarini tartib bilan ko'rish mumkin.

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.address_state,
    co.order_id,
    co.order_date,
    p.product_name,
    p.price
FROM customer c
JOIN customer_order co ON c.customer_id = co.customer_id
JOIN product p ON co.product_id = p.product_id
ORDER BY c.customer_id;
```

Query Result x

SQL | Fetched 50 rows in 0,008 seconds

CUSTOMER_ID	FIRST_NAME	LAST_NAME	ADDRESS_STATE	ORDER_ID	ORDER_DATE	PRODUCT_NAME	PRICE
1	2 Fred	Montgomery	CA		9 23.01.17	Photo Editing Pro	250
2	2 Fred	Montgomery	CA		11 09.06.16	Desk	110,9
3	2 Fred	Montgomery	CA		33 28.10.16	Monitor	149,95
4	2 Fred	Montgomery	CA		46 30.01.16	Chair	79,95
5	3 Lois	Lawson	OR		47 15.12.15	Accounting Plus	60,45

Bu yerda harid qilinsa qilinmasa, barcha haridorlar ko'rsatilyapti:

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.address_state,
    co.order_id,
    co.order_date,
    p.product_name,
    p.price
FROM customer c
LEFT JOIN customer_order co ON c.customer_id = co.customer_id
LEFT JOIN product p ON co.product_id = p.product_id
ORDER BY c.customer_id;
```

Query Result x

SQL | All Rows Fetched: 52 in 0,006 seconds

CUSTOMER_ID	FIRST_NAME	LAST_NAME	ADDRESS_STATE	ORDER_ID	ORDER_DATE	PRODUCT_NAME	PRICE
1	1 Teresa	Hudson	NY	(null)	(null)	(null)	(null)
2	2 Fred	Montgomery	CA		9 23.01.17	Photo Editing Pro	250
3	2 Fred	Montgomery	CA		11 09.06.16	Desk	110,9

Bu yerda department jadvali ham qo'shib chiqarildi:

```

SELECT
c.customer_id,
c.first_name,
c.last_name,
c.address_state,
co.order_id,
co.order_date,
p.product_name,
p.price,
d.department_name
FROM customer c
LEFT JOIN customer_order co ON c.customer_id = co.customer_id
LEFT JOIN product p ON co.product_id = p.product_id
LEFT JOIN department d ON p.department_id = d.department_id
ORDER BY c.customer_id;

```

Script Output x Query Result x

SQL | All Rows Fetched: 52 in 0,008 seconds

	CUSTOMER_ID	FIRST_NAME	LAST_NAME	ADDRESS_STATE	ORDER_ID	ORDER_DATE	PRODUCT_NAME	PRICE	DEPARTMENT_NAME
1	1	Teresa	Hudson	NY	(null)	(null)	(null)	(null)	(null)
2	2	Fred	Montgomery	CA		9 23.01.17	Photo Editing Pro	250	Software Development
3	2	Fred	Montgomery	CA		11 09.06.16	Desk	110,9	Hardware Development

Alternative JOIN syntax

JOINlarning muqobil (o'rindosh yoki sinonim) sintaksisi

Script Output x Query Result x

SQL | All Rows Fetched: 197 in 0,019 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME
1	1	Michelle	Foster	Legal
2	2	Cheryl	Turner	Customer Support
3	3	Carolyn	Hudson	Finance

+ belgisini qo'yishdagi o'zgarish.

O'ng tarafga qo'yilsa, o'ng taraf *NULL*lar bilan to'ldiriladi.

Chap tarafga qo'yilsa, chap taraf *NULL*lar bilan to'ldiriladi.

```
SELECT
employee_id,
first_name,
last_name,
department_name
FROM employee, department
WHERE employee.department_id = department.department_id(+);
```

Script Output | Query Result | SQL | All Rows Fetched: 200 in 0,012 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME
1	84	Jennifer	Long	(null)
2	140	Rebecca	Chapman	(null)
3	114	James	Phillips	(null)
4	119	Paula	Lynch	Software Deve...

```
SELECT
employee_id,
first_name,
last_name,
department_name
FROM employee, department
WHERE employee.department_id(+) = department.department_id;
```

Script Output | Query Result | SQL | All Rows Fetched: 198 in 0,021 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME
1	(null)	(null)	(null)	Maintenance
2	200	Charles	Stone	Software Development
3	199	Lawrence	Henderson	Marketing

Qo'shimcha shartlarni qo'shib ko'ramiz:

```
SELECT
    employee_id,
    first_name,
    last_name,
    department_name,
    salary,
    hire_date
FROM employee, department
WHERE salary > 50000
AND hire_date > '01.01.2012'
AND employee.department_id = department.department_id;
```

Script Output x Query Result x

SQL | Fetched 50 rows in 0,004 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	SALARY	HIRE_DATE
1	2	Cheryl	Turner	Customer Support	79000	02.01.12
2	9	Kathleen	Jones	Finance	92000	15.03.15
3	11	Norma	Henry	Customer Support	56000	24.07.13

Bitta eng kerakli qator bo'lmasligi **ALTERNATIVE JOIN**ni
CARTESIAN JOINga aylantirib yuboradi.

```
SELECT
    employee_id,
    first_name,
    last_name,
    department_name,
    salary,
    hire_date
FROM employee, department
WHERE salary > 50000
AND hire_date > '01.01.2012'
--AND employee.department_id = department.department_id;
;
```

Script Output x Query Result x

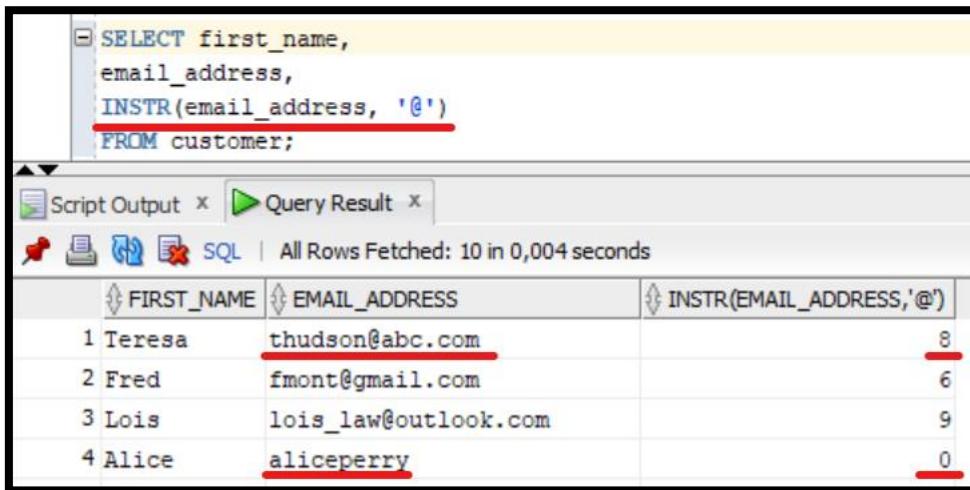
SQL | All Rows Fetched: 891 in 0,069 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	SALARY	HIRE_DATE
1	2	Cheryl	Turner	Executive	79000	02.01.12
2	9	Kathleen	Jones	Executive	92000	15.03.15
3	11	Norma	Henry	Executive	56000	24.07.13

FUNCTIONS

String functions

INSTR(matn, qidirilayotgan belgi) — belgini qidirib, uning tartib raqamini chiqaradi.



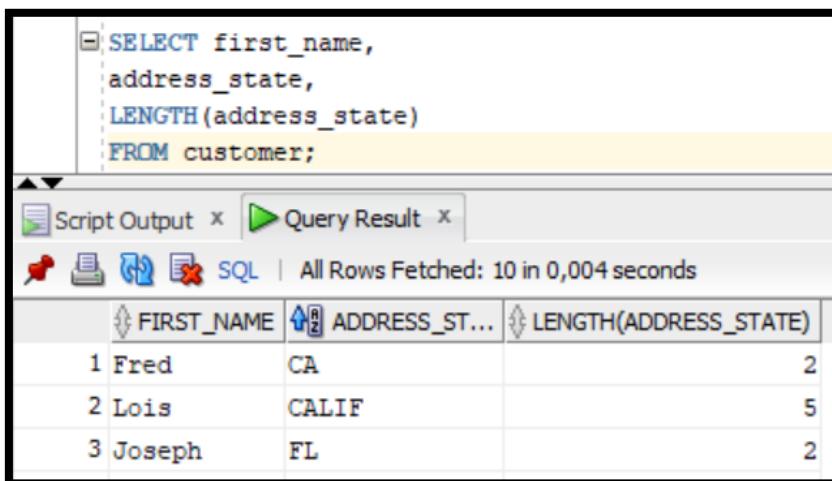
The screenshot shows an Oracle SQL Developer interface. The SQL editor contains the following query:

```
SELECT first_name,
       email_address,
       INSTR(email_address, '@')
  FROM customer;
```

The results are displayed in a table titled "Query Result". The columns are FIRST_NAME, EMAIL_ADDRESS, and INSTR(EMAIL_ADDRESS, '@'). The data is as follows:

FIRST_NAME	EMAIL_ADDRESS	INSTR(EMAIL_ADDRESS, '@')
1 Teresa	thudson@abc.com	8
2 Fred	fmont@gmail.com	6
3 Lois	lois_law@outlook.com	9
4 Alice	aliceperry	0

LENGTH(matn) — belgilar sonini hisoblaydi.



The screenshot shows an Oracle SQL Developer interface. The SQL editor contains the following query:

```
SELECT first_name,
       address_state,
       LENGTH(address_state)
  FROM customer;
```

The results are displayed in a table titled "Query Result". The columns are FIRST_NAME, ADDRESS_ST..., and LENGTH(ADDRESS_STATE). The data is as follows:

FIRST_NAME	ADDRESS_ST...	LENGTH(ADDRESS_STATE)
1 Fred	CA	2
2 Lois	CALIF	5
3 Joseph	FL	2

INSTR va LENGTH aniq qiymat qaytargani uchun raqamli shartga kiritish mumkin:

```

SELECT first_name,
       email_address,
       INSTR(email_address, '@')
       address_state,
       LENGTH(address_state)
  FROM customer
 WHERE INSTR(email_address, '@') > 0
   AND LENGTH(address_state) = 2
 ORDER BY LENGTH(address_state) DESC;

```

Script Output X Query Result X

All Rows Fetched: 7 in 0,005 seconds

FIRST_NAME	EMAIL_ADDRESS	ADDRESS_STATE	LENGTH(ADDRESS_STATE)
1 Ralph	ralph_mont25@gmail.com	13	2
2 Fred	fmont@gmail.com	6	2
3 Lois	lois_law@outlook.com	9	2

Nesting functions within functions

Nesting functions — funksiyalarni kombinatsion ishlatalish.

SUBSTR(matn, n-harfdan boshlab, m ta harf chiqarilsin) — Satrdan pastki qatorni ajratib oladi (5-pozitsiyadan boshlab, 3 ta belgini ajratib ol):

`SELECT SUBSTR("SqlTutorial", 5, 3) ;`

Natija: uto

@ belgisidan boshlab nechta belgi borligini aniqlash:

```

SELECT first_name,
       email_address,
       INSTR(email_address, '@'),
       SUBSTR(email_address, INSTR(email_address, '@'), LENGTH(email_address))
  FROM customer;

```

Script Output X Query Result X

All Rows Fetched: 10 in 0,002 seconds

FIRST_NAME	EMAIL_ADDRESS	INSTR(EMAIL_ADDRESS,'@')	SUBSTR(EMAIL_A
1 Teresa	thudson@abc.com	8	@abc.com
2 Fred	fmont@gmail.com	6	@gmail.com
3 Lois	lois_law@outlook.com	9	@outlook.com

Bunda *SUBSTR*dan foydalanib, *email_address* ustuni tanlandi, *INSTR* orqali @ belgisi aniqlandi, *LENGTH* orqali @ belgisi davomidagi belgilar ajratib olindi.

@ belgisidan keyingi belgilarni aniqlash. Ya’ni elektron pochta domenini aniqlash:

The screenshot shows a SQL query in the editor and its execution results in the query output window. The query uses *INSTR* to find the position of '@' in the *email_address* column and *SUBSTR* to extract the domain part starting from that position plus one character to the end of the string. The results show three rows with columns: FIRST_NAME, EMAIL_ADDRESS, INSTR(EMAIL_ADDRESS, '@'), and EMAIL_DOMAIN. The extracted domains are abc.com, gmail.com, and outlook.com respectively.

```
SELECT first_name,
       email_address,
       INSTR(email_address, '@'),
       SUBSTR(email_address, INSTR(email_address, '@')+1, LENGTH(email_address)) AS email_domain
  FROM customer;
```

FIRST_NAME	EMAIL_ADDRESS	INSTR(EMAIL_ADDRESS, '@')	EMAIL_DOMAIN
1 Teresa	thudson@abc.com	8	abc.com
2 Fred	fmont@gmail.com	6	gmail.com
3 Lois	lois_law@outlook.com	9	outlook.com

Number Function

ROUND(number, raqam [10 lik qismigacha]) — sonni yaxlitlaydi.

CEIL(number) — sonni kattasi tomonga yaxlitlaydi.

FLOOR(number) — faqat butun qismini oladi.

The screenshot shows a query in the editor and its execution results in the query output window. The query demonstrates the use of *ROUND*, *CEIL*, and *FLOOR* functions on the *price* column of the *product* table. It includes comments explaining the purpose of each function: *ROUND* for rounding to the nearest integer, *ROUND(price, 1)* for rounding to one decimal place, *CEIL* for rounding up to the next integer, and *FLOOR* for rounding down to the previous integer. The results show two products with their original price, rounded price, and the results of the three functions.

```
SELECT
       product_name,
       price,
       ROUND(price),    -- butun qismni yaxlitlaydi
       ROUND(price, 1),  -- 10 lik qismgacha yaxlitlaydi
       CEIL(price),     -- kattalashtiradi
       FLOOR(price) AS "FLOOR" -- kichiklashtiradi
      FROM product
     WHERE product_id IN(5,6);
```

PRODUCT_NAME	PRICE	ROUND(PRICE)	ROUND(PRICE,1)	CEIL(PRICE)	FLOOR
1 Microsoft Office	121,15	121	121,2	122	121
2 Antivirus Extreme	48,5	49	48,5	49	48

Misol: 9 ta Monitor harid qilingani uchun 85% chegirma berildi. Uni hisoblash quyidagicha turda bo’lishi mumkin:

```

SELECT
product_name,
price,
price*9*0.85 AS NARX,
ROUND(price*9*0.85),
ROUND(price*9*0.85, 2),
CEIL(price*9*0.85),
FLOOR(price*9*0.85)
FROM product
WHERE product_id = 1;

```

Query Result x

SQL | All Rows Fetched: 1 in 0,006 seconds

PRODUCT_NAME	PRICE	NARX	ROUND(PRICE*9*0.85)	ROUND(PRICE*9*0.85,2)	CEIL(PRICE*9*0.85)	FLOOR(PRICE*9*0.85)
1 Monitor	149,95	1147,1175	1147	1147,12	1148	1147

Date Function

SYSDATE — hozirgi kun sanasi

ADD_MONTHS(sana, oylar_soni) — falon sanaga, falon oy qo'shish

MONTHS_BETWEEN(sanadan, sanagacha) — sanalar orasidagi oylarni chiqarib beradi.

```

SELECT
first_name,
last_name,
hire_date,
SYSDATE,
ADD_MONTHS(hire_date, 12) AS rewiev_date,
ROUND(MONTHS_BETWEEN(SYSDATE, hire_date)/12, 1) AS years_with_company
FROM employee;

```

Query Result x

SQL | Fetched 50 rows in 0,008 seconds

	FIRST_NAME	LAST_NAME	HIRE_DATE	SYSDATE	REWIEV_DATE	YEARS_WITH_COMPANY
1	Michelle	Foster	27.08.11	01.03.23	27.08.12	11,5
2	Cheryl	Turner	02.01.12	01.03.23	02.01.13	11,2
3	Carolyn	Hudson	04.12.16	01.03.23	04.12.17	6,2

Soddaroq misol:

```

SELECT
    SYSDATE,
    ADD_MONTHS(SYSDATE, 60),
    MONTHS_BETWEEN(ADD_MONTHS(SYSDATE, 60), SYSDATE)/12 AS yillar_farqi
FROM dual;

```

Query Result | All Rows Fetched: 1 in 0,004 seconds

SYSDATE	ADD_MONTHS(SYSDATE,60)	YILLAR_FARQI
01.03.23	01.03.28	5

Data Types and Conversion Function

- CHAR: character string with fixed size
- VARCHAR2: character string with a variable size
- NUMBER: stores numeric data with optional decimals
- DATE: stores date and time
- TIMESTAMP: stores date, time, and fractional seconds
- CLOB: stores large amounts of text

Converting

- TO_CHAR, TO_DATE, TO_NUMBER

*TO_CHAR*ga misol:

```

SELECT
    first_name,
    last_name,
    hire_date,
    TO_CHAR(hire_date, 'YYYY_MM_DD')
FROM employee;

```

Query Result | Fetched 50 rows in 0,004 seconds

FIRST_NAME	LAST_NAME	HIRE_DATE	TO_CHAR(HIRE_DATE,'YYYY_MM_DD')
1 Michelle	Foster	27.08.11	2011_08_27
2 Cheryl	Turner	02.01.12	2012_01_02
3 Carolyn	Hudson	04.12.16	2016_12_04

TO_CHAR, *TO_DATE* va *TO_NUMBER*ga misol:

```
SELECT
    TO_CHAR(SYSDATE,'DD_MM_YYYY') AS BUGUN,
    TO_DATE('12_11_2010', 'DD_MM_YYYY') AS SANA,
    TO_NUMBER('200') AS NOMER
FROM dual;
```

Query Result | All Rows Fetched: 1 in 0,005 seconds

BUGUN	SANA	NOMER
1 01_03_2023	12.11.10	200

Case Statement

```
CASE [expression] WHEN condition_1 THEN result_1
WHEN condition_2 THEN result_2
...
WHEN condition_n THEN result_n
ELSE result
END case_name
```

Misol:

```
SELECT
    product_id,
    product_name,
    price,
    CASE
        WHEN price > 100 THEN 'Over 100'
        WHEN price <= 100 THEN 'Less than or under 100'
    END price_group
FROM product;
```

Query Result | All Rows Fetched: 8 in 0,006 seconds

PRODUCT_ID	PRODUCT_NAME	PRICE	PRICE_GROUP
1	1 Monitor	149,95	Over 100
2	2 Desk	110,9	Over 100
3	3 Chair	79,95	Less than or under 100

Murakkabroq misol:

```
SELECT
product_id,
product_name,
price,
CASE
WHEN price > 200 THEN 'Over 200'
WHEN price > 100 AND price <= 200 THEN 'Between 100 and 200'
WHEN price > 50 AND price <= 100 THEN 'Between 50 and 100'
ELSE 'Under 50'
END price_group
FROM product;
```

Query Result | All Rows Fetched: 8 in 0,006 seconds

	PRODUCT_ID	PRODUCT_NAME	PRICE	PRICE_GROUP
4	4	Accounting Plus	60,45	Between 50 and 100
5	5	Microsoft Office	121,15	Between 100 and 200
6	6	Antivirus Extreme	48,5	Under 50
7	7	Photo Editing Pro	250	Over 200

Address_state bo'yicha Caseni qo'llash:

```
SELECT
first_name,
address_state,
CASE address_state
WHEN 'NY' THEN 'East'
WHEN 'CA' THEN 'West'
WHEN 'OR' THEN 'West'
WHEN 'SC' THEN 'East'
WHEN 'TX' THEN 'West'
WHEN 'FL' THEN 'East'
WHEN 'IN' THEN 'East'
ELSE 'Unknown'
END state_group
FROM customer;
```

Query Result | All Rows Fetched: 10 in 0,005 seconds

	FIRST_NAME	ADDRESS_STATE	STATE_GR...
5	Teresa	NY	East
6	Lois	CALIF	Unknown
7	Dorothy	OR	West

Huddi shunga sinonim va optimal kod:

```

SELECT
    first_name,
    address_state,
    CASE
        WHEN address_state IN ('NY', 'SC', 'FL', 'IN') THEN 'East'
        WHEN address_state IN ('CA', 'OR', 'TX') THEN 'West'
        ELSE 'Unknown'
    END state_group
FROM customer;

```

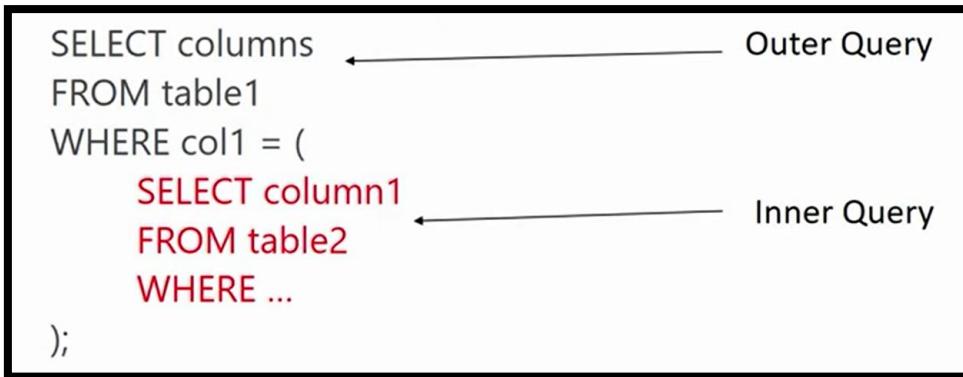
Query Result | All Rows Fetched: 10 in 0,023 seconds

FIRST_NAME	ADDRESS_STATE	STATE_GR...
5 Teresa	NY	East
6 Lois	CALIF	Unknown
7 Dorothy	OR	West

Subqueries

About

So'rov osti so'rovlar



Types:

Single row subquery: returns one row (1 qatorli ichki so'rov)

Multi row subquery: returns multiple rows (ko'p qatorli ichki so'rov)

Single row subquery

Examples: AVG() ya'ni sonlarning o'rtachasi yagona qiymat hisoblanadi. Shuning uchun u Single row subquery

Ichki so'rov: maoshlarning o'rtacha qiymati aniqlab olindi.

```
SELECT AVG(salary)
FROM employee;
```

Tashqi so'rov + ichki so'rov: o'rtacha maoshdan yuqori maosh oluvchilar chiqarildi.

```
2 SELECT
employee_id,
first_name,
last_name,
salary
FROM employee
WHERE salary > (
    1 SELECT AVG(salary)
    FROM employee
);

```

Query Result x

SQL | Fetched 50 rows in 0,007 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SAL...
1	126	Christopher	Lawrence	69000
2	45	Lori	Jacobs	69000
3	109	Paula	Dunn	69000
4	98	James	Rodriguez	70000

Multi row subquery

Ko'p qatorli ichki so'rovlar quyidaqilar bilan ishlataladi:

- E.g. IN(), > ANY, < ALL

Department name : Sales va Finance bo'lqanlarni chigарib oldik.

```

SELECT department_id,
       department_name
  FROM department
 WHERE department_name IN('Sales', 'Finance');

```

Query Result | All Rows Fetched: 2 in 0,002 seconds

DEPARTMENT_ID	DEPARTMENT_NAME
1	Sales
2	Finance

Tashqi + ichki so'rovlar jamlandi:

```

SELECT
    employee_id,
    first_name,
    last_name,
    department_id
   FROM employee
  WHERE department_id IN(
      SELECT department_id
        FROM department
       WHERE department_name IN('Sales', 'Finance')
  );

```

Query Result | Fetched 50 rows in 0,009 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_ID
10	44 Philip	Jordan	7
11	45 Lori	Jacobs	2
12	49 Melissa	Cunningham	2
13	51 Martha	Cruz	7

Inserting, Updating an Deleting Data

Inserting Data

Ma'lumotni bittalab kiritish:

```
INSERT INTO table_name [(col1, col2...)]
```

```
VALUES (value1, value2);
```

Ma'lumot kiritdik:

```
INSERT INTO employee(employee_id, first_name, last_name, department_id)
VALUES (300, 'John', 'Smith', 3);
```

Script Output X | Query Result X
| Task completed in 0,039 seconds
1 row inserted.

```
INSERT INTO employee(employee_id, first_name, last_name, department_id, salary, manager_id, hire_date)
VALUES (301, 'Marge', 'Abbot', 2, 31000, 51, '04.01.2017');
```

Script Output X | Query Result X
| Task completed in 0,051 seconds
1 row inserted.

Ma'lumotni birdaniga ko'p kiritish uchun shablon:

```
INSERT ALL
INTO table_namel(coll, col2) VALUES (val1, val2)
INTO table_name2(coll, col2) VALUES (val1, val2);
```

Misol:

```
INSERT ALL
INTO employee(employee_id, first_name, last_name, department_id) VALUES (303, 'Mark', 'Spencer', 4)
INTO employee(employee_id, first_name, last_name, department_id) VALUES (304, 'Simone', 'Fletcher', 3)
INTO employee(employee_id, first_name, last_name, department_id) VALUES (305, 'Alison', 'Smith', 8)
SELECT * FROM dual;
```

Script Output X | Query Result X
| Task completed in 0,04 seconds
3 rows inserted.

Pastdagi natijani tozalaydi: Ctrl + Shift + D

Script Output X
| Task completed in 0,041 seconds
Error Clear (Ctrl+Shift+D)
SQL Error: ORA-00905: отсутствует ключевое слово
00905. 00000 - "missing keyword"

Inserting Data From Another Table

Boshqa jadvaldan ma'lumot qo'shish

- `INSERT INTO target_table (columns)
SELECT (columns) FROM source_table...`

Misol. Customer jadvalidagi id=1 bo'lgan shaxsni ko'ramiz.

The screenshot shows a MySQL Workbench interface. In the SQL editor, the following query is written:

```
SELECT first_name, last_name  
FROM customer  
WHERE customer_id = 1;
```

In the Query Result tab, the output is:

	FIRST_NAME	LAST_NAME
1	Teresa	Hudson

Ushbu shaxsni employee jadvaliga kiritamiz:

The screenshot shows a MySQL Workbench interface. In the SQL editor, the following query is written:

```
INSERT INTO employee(employee_id, first_name, last_name)  
SELECT 250, first_name, last_name  
FROM customer  
WHERE customer_id = 1;
```

In the Query Result tab, the output is:

1 row inserted.

Id NOT NULL bo'lgani uchun 250 deb kiritdik.

Updating Data

About

Ma'lumotlarni yangilash:

- `UPDATE table
SET col1 = val1,
 col2 = val2
[WHERE condition]`

Bitta qator ma'lumotini o'zgartirish

Employee_id = 85 bo'lgan shaxsni chiqaramiz:

The screenshot shows a MySQL Workbench interface. On the left, a SQL editor window contains the following query:

```
SELECT*
FROM employee
WHERE employee_id = 85;
```

On the right, a "Query Result" window displays the results:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY
1	85	Johnny	Reed	22000

Uni oyligini 10.000 ga oshiramiz:

The screenshot shows two MySQL Workbench windows. The left window is a "Script Output" window showing the execution of an UPDATE statement:

```
UPDATE employee
SET salary = salary + 10000
WHERE employee_id = 85;
```

The right window is a "Query Result" window showing the updated data:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY
1	85	Johnny	Reed	32000

Ko'p qator ma'lumotlarini o'zgartirish

The screenshot shows a MySQL Workbench interface. A "Script Output" window contains the following query:

```
SELECT*
FROM employee
WHERE employee_id IN (102, 59, 16);
```

A "Query Result" window shows the results:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	MANAGER_ID
1	16	Fred	Nichols	70000	3	186
2	59	Christine	Brooks	40000	6	57
3	102	Ralph	Woods	22000	4	10

The screenshot shows a MySQL Workbench interface. A "Script Output" window contains the following query:

```
UPDATE employee
SET manager_id = 30, salary = salary + 5000
WHERE employee_id IN (102, 59, 16);
```

A "Query Result" window shows the results:

3 rows updated.

```
SELECT*
FROM employee
WHERE employee_id IN (102, 59, 16);
```

Script Output x Query Result x
SQL | All Rows Fetched: 3 in 0,004 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	DEPARTMENT_ID	MANAGER_ID
1	16	Fred	Nichols	75000	3	30
2	59	Christine	Brooks	45000	6	30
3	102	Ralph	Woods	27000	4	30

Deleting Data

Ma'lumotlarni o'chirish.

O'chiriladigan ma'lumotni doim tekshirib olish kerak.

```
SELECT* FROM employee
WHERE employee_id = 250;
```

Script Output x Query Result x
SQL | All Rows Fetched: 1 in 0,004 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME
1	250	Teresa	Hudson

Endi uni o'chiramiz:

O'chirishda WHERE bandi borligiga ishonch hosil qiling.

Aks holda barcha ma'lumotlar o'chib ketadi.

```
DELETE FROM employee
WHERE employee_id = 250;
```

Script Output x Query Result x
SQL | Task completed in 0,0

1 row deleted.

Tekshiramiz:

```
SELECT * FROM employee  
WHERE employee_id = 250;
```

Script Output x Query Result x
SQL | All Rows Fetched: 0 in 0,002

EMPLOYEE_ID	FIRST_NAME	LAST_NAME

Endi u ma'lumot mavjud emas.

Commit and Rollback

COMMIT — yangilanishni saqlash.

ROLLBACK — avvalgi holatiga qaytarish.

Ma'lumot kiritdik:

```
INSERT INTO product (product_id, product_name, price, department_id)  
VALUES (12, 'Large Table', 220.50, 2);
```

Script Output x Query Result x
SQL | Task completed in 0,032 seconds

1 row inserted.

Tekshiramiz:

```
SELECT * FROM product;
```

Script Output x Query Result x
SQL | All Rows Fetched: 9 in 0,006 seconds

PRODUCT_ID	PRODUCT_NAME	PRICE	DEPARTMENT_ID
1	12 Large Table	220,5	2

COMMIT: saqlash

```
COMMIT;
```

Script Output x Query Result x
SQL | Task completed in 0,001 seconds

Commit complete.

Yana ma'lumot kiritamiz:

```
INSERT INTO product (product_id, product_name, price, department_id)
VALUES (15, 'Red Chair', 52, 6);
```

Script Output x Query Result x
Task completed in 0,036 seconds
1 row inserted.

Tekshiramiz:

```
SELECT*FROM product;
```

Script Output x Query Result x
SQL | All Rows Fetched: 10 in 0,003 seconds

	PRODUCT_ID	PRODUCT_NAME	PRICE	DEPARTMENT_ID
1	15	Red Chair	52	6

Rollback: saqlashdan boshlab ortga qaytarish

```
ROLLBACK;
```

Script Output x Query Result x
Rollback complete.

Tekshiramiz:

```
SELECT*FROM product;
```

Script Output x Query Result x
SQL | All Rows Fetched: 9 in 0,003 seconds

	PRODUCT_ID	PRODUCT_NAME	PRICE	DEPARTMENT_ID
1	12	Large Table	220,5	2

Natija: yangilanish avvalgi holatiga qaytarildi.

Truncating Data

Ma'lumotlarni qaytarib bo'lmaydigan qilib o'chirish.

Customer_order jadvalini chiqaramiz:

```
SELECT * FROM customer_order;
```

Script Output x Query Result x
SQL | Fetched 50 rows in 0,01 seconds

ORDER_ID	CUSTOMER_ID	PRODUCT_ID	ORDER_DATE
1	1	3	2 18.12.16
2	2	5	6 01.06.15
3	3	6	5 26.09.16

TRUNCATE TABLE

```
TRUNCATE TABLE customer_order;
```

Query Result x Script Output x
SQL | Task completed in 0.031 seconds

Table CUSTOMER_ORDER truncated.

Tekshiramiz:

```
SELECT * FROM customer_order;
```

Script Output x Query Result x
SQL | All Rows Fetched: 0 in 0.002 seconds

ORDER_ID	CUSTOME...	PRODUCT...	ORDER_D...
----------	------------	------------	------------

ROLLBACK:

```
ROLLBACK;
```

Script Output x Query Result x
SQL | Task completed in 0.001 seconds

Table CUSTOMER_ORDER truncated.

Rollback complete.

Yana tekshiramiz:

```

SELECT * FROM customer_order;

```

Script Output x Query Result x

SQL | All Rows Fetched: 0 in 0.002 seconds

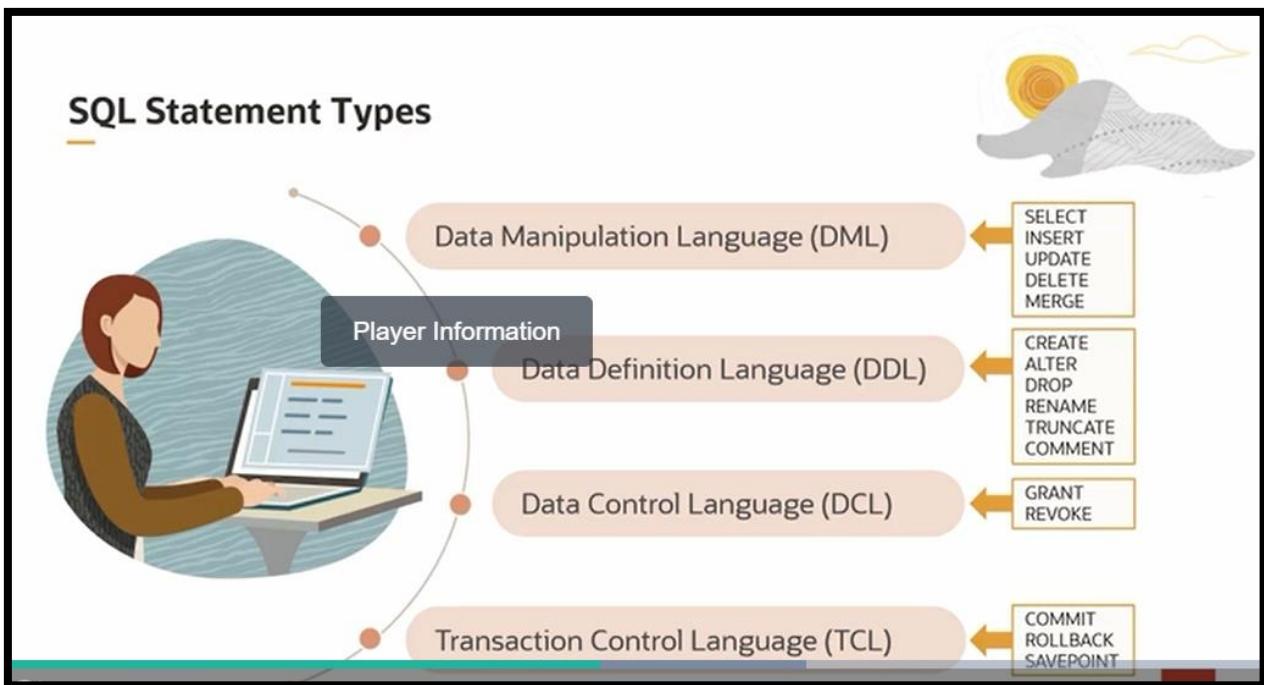
ORDER_ID	CUSTOMER_ID	PRODUCT_ID	ORDER_DATE
----------	-------------	------------	------------

Yangilanish avvalgi holatiga qaytarilmadi.

Shuning uchun TRUNCATEni ishlatalish havfli hisoblanadi.

Creating, Altering and Dropping Tables

SQL Statement Types



Create a Table

```

CREATE TABLE tablename (
    colname data_type constraints
    ...
);

```

Ma'lumot turi va ustun nomlari 30 belgidan oshmasligi kerak.

Ma'lumot turlari:

VARCHAR2

VARCHAR2(50) — 50 belgi uchun joy ajratadi

NUMBER

NUMBER(3) = XXX

NUMBER(4,2) = XX.XX

[4- raqamlarning umumiyligi soni, 2 – 10 liklar soni]

Yangi jadval yaratamiz:

The screenshot shows the SQL script editor with the following code:

```
CREATE TABLE job_role (
    job_role_id NUMBER(10),
    role_name VARCHAR2(50),
    role_create_date DATE
);
```

Below the code, the status bar indicates "Task completed in" and the message "Table JOB_ROLE created." is displayed.

Tekshiramiz:

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

```
SELECT * FROM job_role;
```

Below the query, the status bar indicates "All Rows Fetched: 0 in 0,007 s". The results pane shows the structure of the table:

JOB_ROL...	ROLE_NAME	ROLE_CR...
------------	-----------	------------

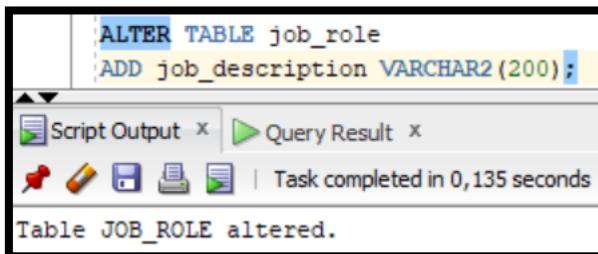
Jadval yaratildi, lekin hali hech qanday ma'lumot kiritilmagan.

Alter Table

- Add column
- Change data type of column
- Add constraint (cheklov)
- Remove / Rename column
- Rename a table

Add Column

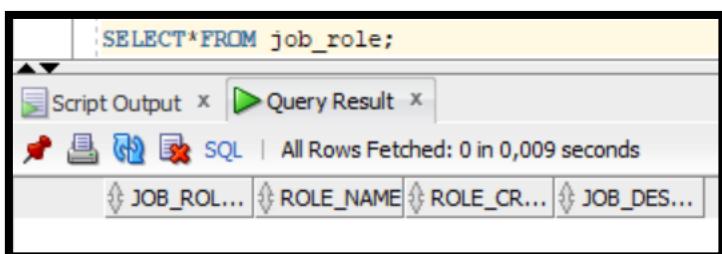
Avvalgi yaratgan bazamizga ustun qo'shdik:



```
ALTER TABLE job_role
ADD job_description VARCHAR2(200);
```

Script Output X | Query Result X
Task completed in 0,135 seconds
Table JOB_ROLE altered.

Tekshiramiz:

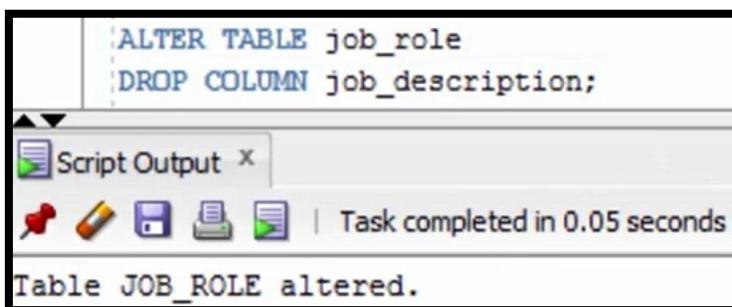


```
SELECT*FROM job_role;
```

Script Output X | Query Result X
SQL | All Rows Fetched: 0 in 0,009 seconds
JOB_ROL... | ROLE_NAME | ROLE_CR... | JOB_DES...

Drop Column

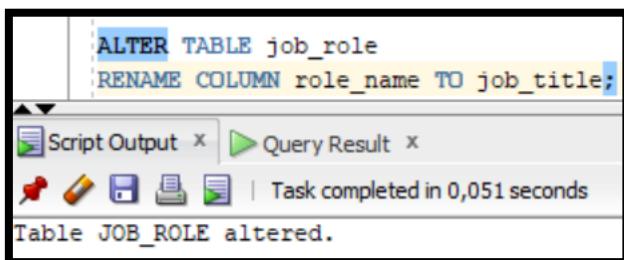
Ustunni o'chiramiz: DROP COLUMN



```
ALTER TABLE job_role
DROP COLUMN job_description;
```

Script Output X
Task completed in 0.05 seconds
Table JOB_ROLE altered.

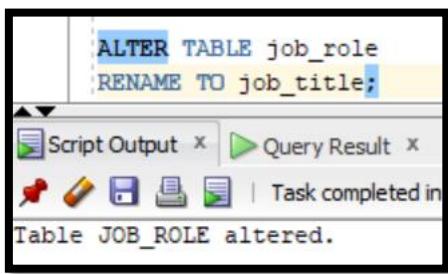
Rename Column



```
ALTER TABLE job_role
RENAME COLUMN role_name TO job_title;
```

Script Output X | Query Result X
Task completed in 0,051 seconds
Table JOB_ROLE altered.

Rename Table



The screenshot shows a SQL query window with the following content:

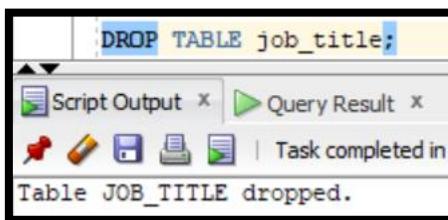
```
ALTER TABLE job_role  
RENAME TO job_title;
```

Below the query window, the status bar displays:

Script Output X | Query Result X
| Task completed in
Table JOB_ROLE altered.

Drop Table

DROP – faqat obyekt uchun ishlataladi.



The screenshot shows a SQL query window with the following content:

```
DROP TABLE job_title;
```

Below the query window, the status bar displays:

Script Output X | Query Result X
| Task completed in
Table JOB_TITLE dropped.