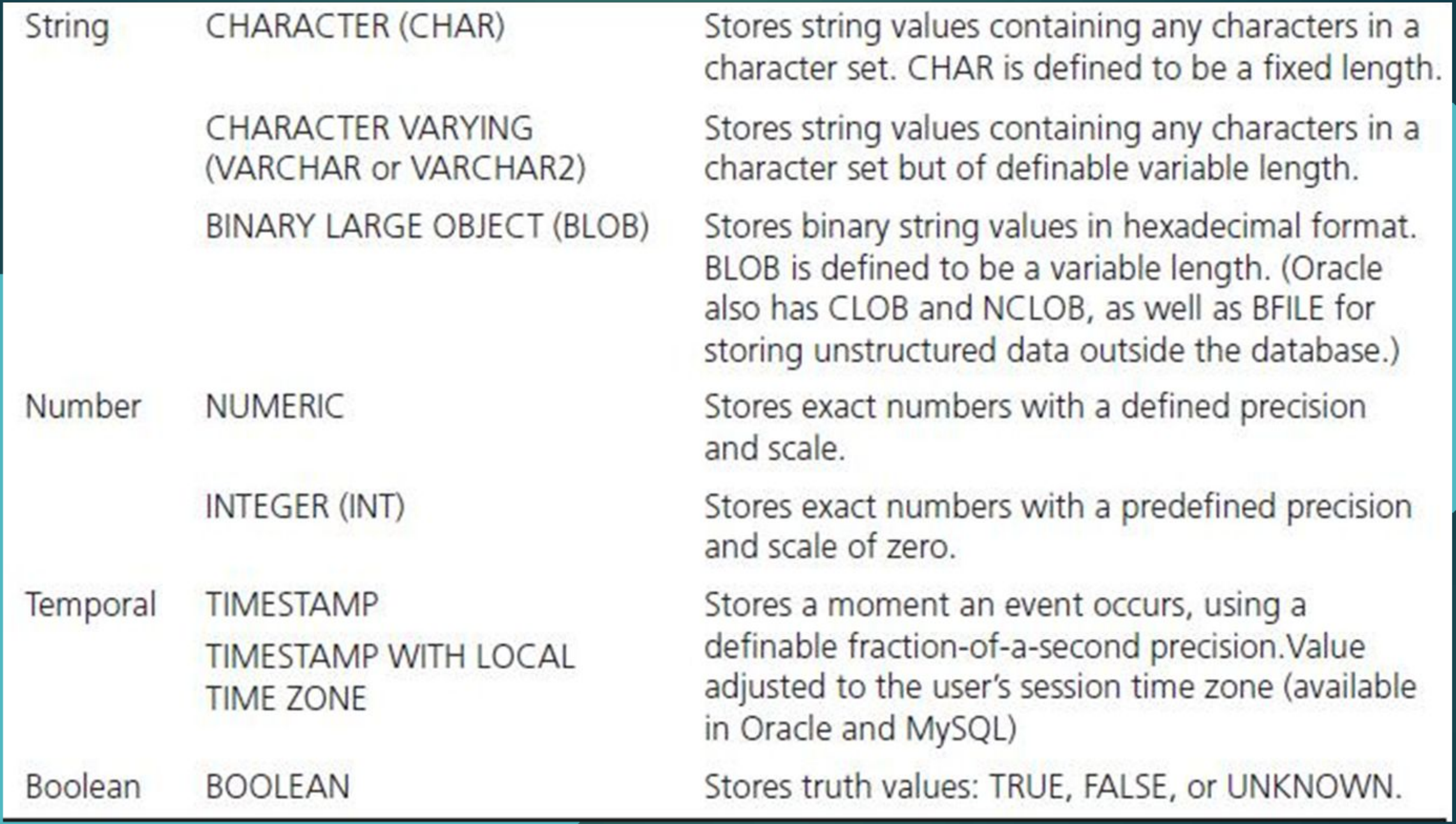
Introduction to SQL

What is SQL?

- **Structured Query Language**
- SQL is Structured Query Language, which is a computer language for **storing**, **manipulating** and **retrieving** data stored in relational database.

- SQL is the standard language for **Relation** Database System. All relational database management systems like “MySQL, MS Access, Oracle, Sybase, Informix, postgres and SQL Server” use SQL as standard database language.

SQL DATA TYPES

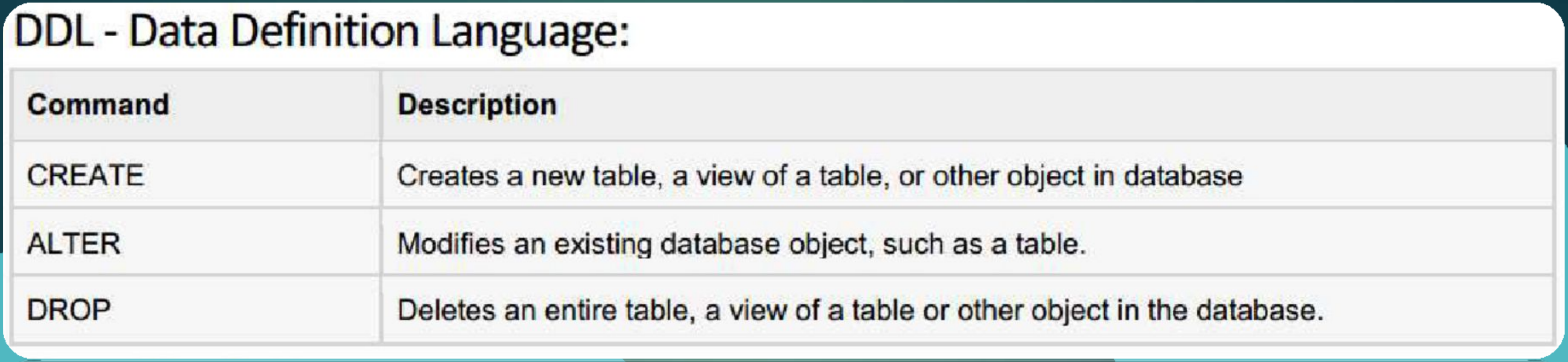


String	CHARACTER (CHAR)	Stores string values containing any characters in a character set. CHAR is defined to be a fixed length.
	CHARACTER VARYING (VARCHAR or VARCHAR2)	Stores string values containing any characters in a character set but of definable variable length.
	BINARY LARGE OBJECT (BLOB)	Stores binary string values in hexadecimal format. BLOB is defined to be a variable length. (Oracle also has CLOB and NCLOB, as well as BFILE for storing unstructured data outside the database.)
Number	NUMERIC	Stores exact numbers with a defined precision and scale.
	INTEGER (INT)	Stores exact numbers with a predefined precision and scale of zero.
Temporal	TIMESTAMP	Stores a moment an event occurs, using a definable fraction-of-a-second precision.
	TIMESTAMP WITH LOCAL TIME ZONE	Value adjusted to the user's session time zone (available in Oracle and MySQL)
Boolean	BOOLEAN	Stores truth values: TRUE, FALSE, or UNKNOWN.

SQL Commands

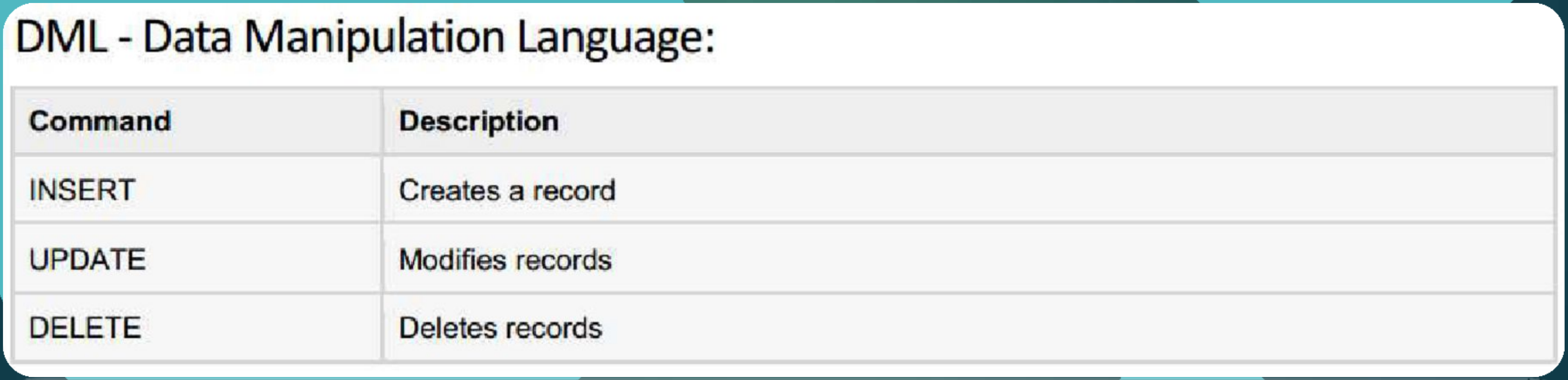
- DDL - **Data Definition Language**
- DML - **Data Manipulation Language**
- DCL - **Data Control Language**
- DQL - **Data Query Language**

SQL Commands



DDL - Data Definition Language:

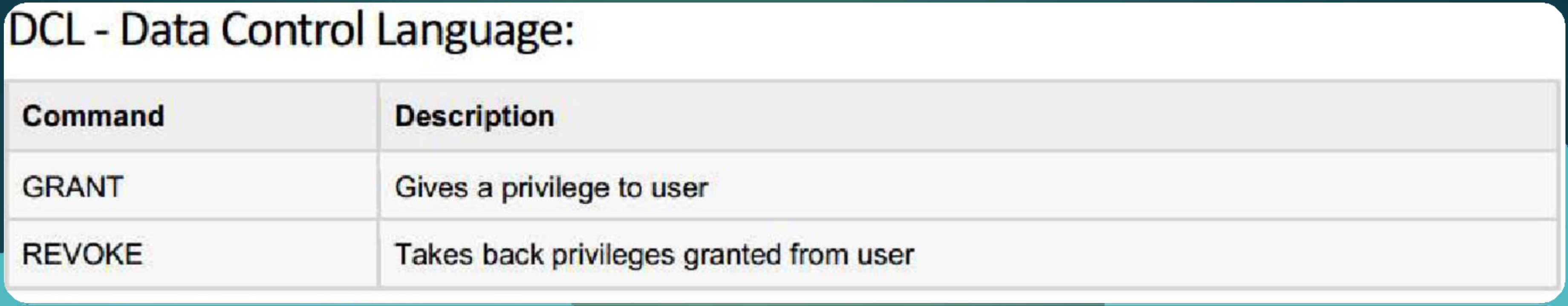
Command	Description
CREATE	Creates a new table, a view of a table, or other object in database
ALTER	Modifies an existing database object, such as a table.
DROP	Deletes an entire table, a view of a table or other object in the database.



DML - Data Manipulation Language:

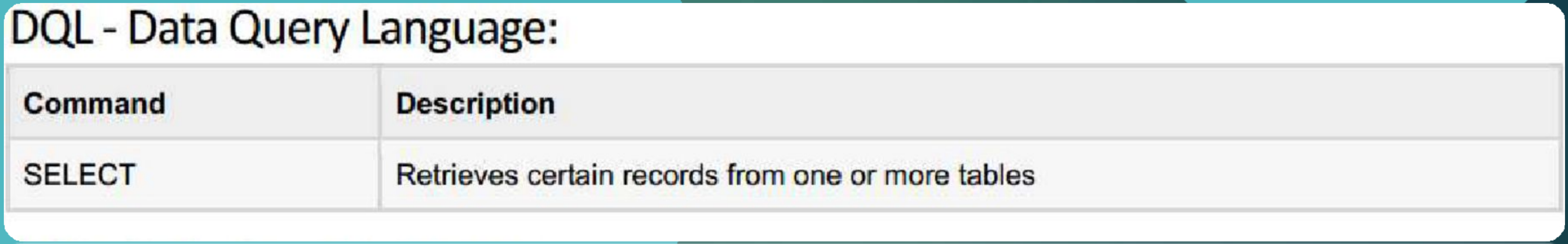
Command	Description
INSERT	Creates a record
UPDATE	Modifies records
DELETE	Deletes records

SQL Commands



DCL - Data Control Language:

Command	Description
GRANT	Gives a privilege to user
REVOKE	Takes back privileges granted from user

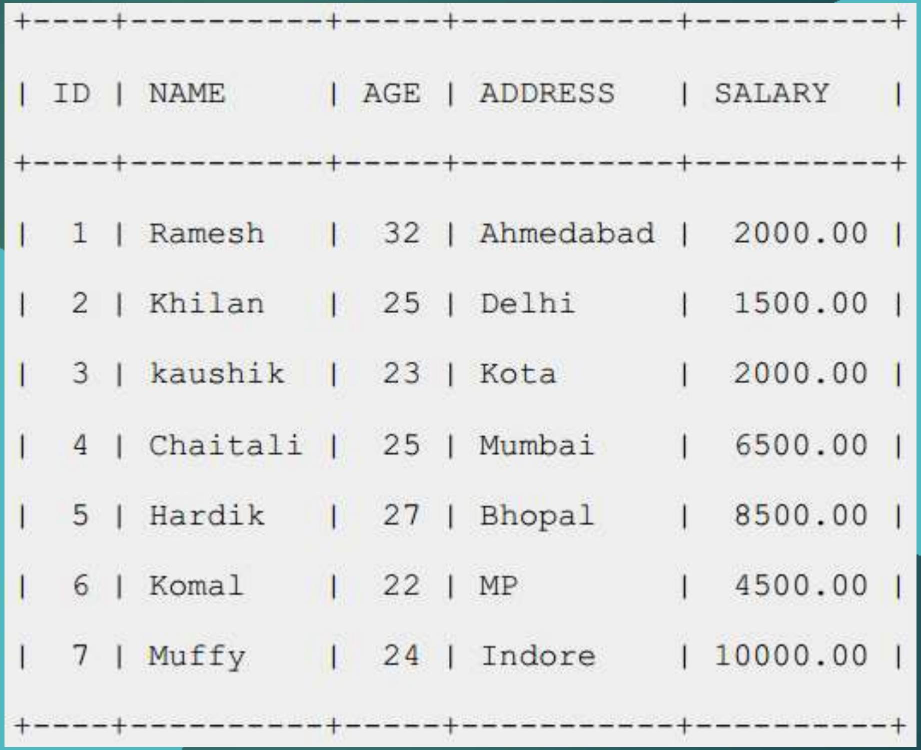


DQL - Data Query Language:

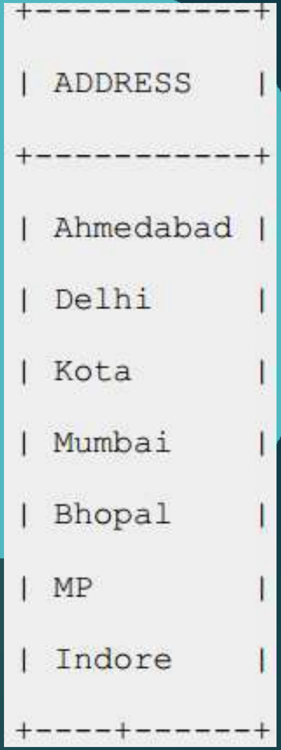
Command	Description
SELECT	Retrieves certain records from one or more tables

SQL RDBMS Concepts

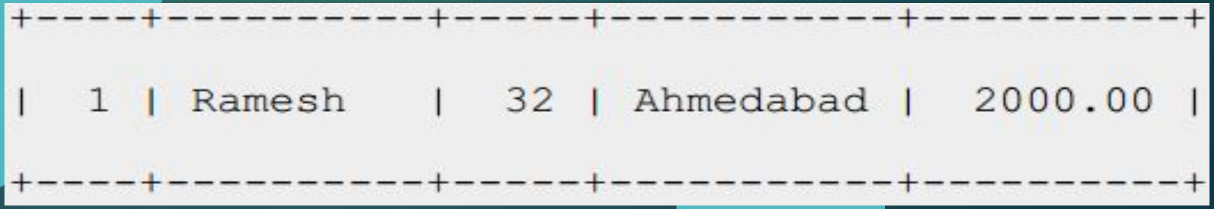
- TABLE
  - RECORD
  - COLUMN
  - CELL



```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```



```
+-----------+
| ADDRESS   |
+-----------+
| Ahmedabad |
| Delhi     |
| Kota      |
| Mumbai    |
| Bhopal    |
| MP        |
| Indore    |
+----+------+
```



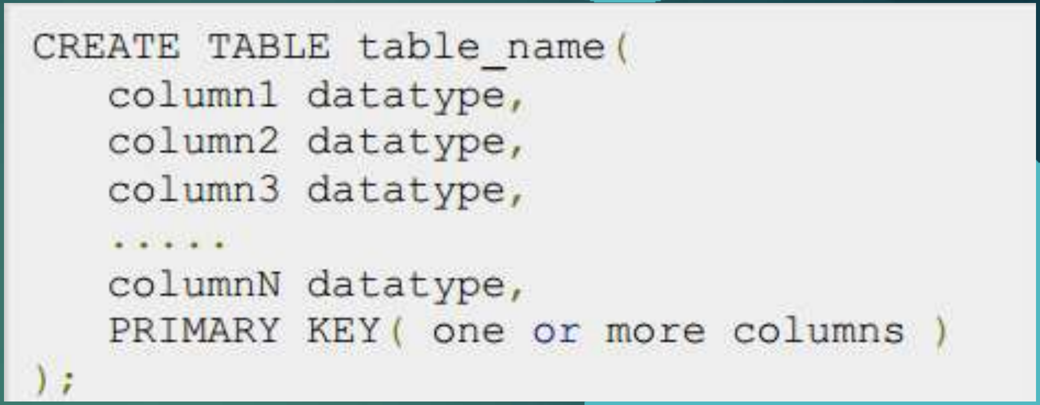
```
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
+----+----------+-----+-----------+----------+
```

SQL RDBMS Concepts

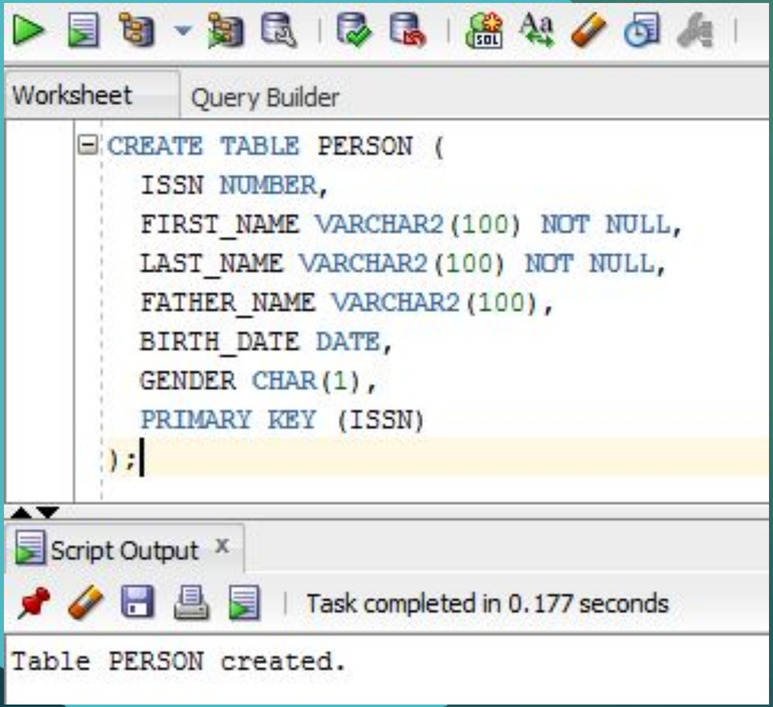
- SQL Constraints: (applied on columns)
  - NOT NULL Constraint
  - UNIQUE Constraint
  - PRIMARY Key
  - FOREIGN Key
  - CHECK Constraint
- Data Integrity:
  - Entity Integrity: There are no duplicate rows in a table
  - Domain Integrity: Enforces valid entries for a given column by Restricting the type by
  - Referential Integrity: Rows cannot be deleted which are used other records
  - User-Defined Integrity: Enforces some specific business rules

SQL : DDL

- Create Table:
  - COLUMNS
  - CONSTRAINTS



```
CREATE TABLE table_name(
   column1 datatype,
   column2 datatype,
   column3 datatype,
   .....
   columnN datatype,
   PRIMARY KEY( one or more columns )
);
```



```
CREATE TABLE PERSON (
  ISSN NUMBER,
  FIRST_NAME VARCHAR2(100) NOT NULL,
  LAST_NAME VARCHAR2(100) NOT NULL,
  FATHER_NAME VARCHAR2(100),
  BIRTH_DATE DATE,
  GENDER CHAR(1),
  PRIMARY KEY (ISSN)
);
```

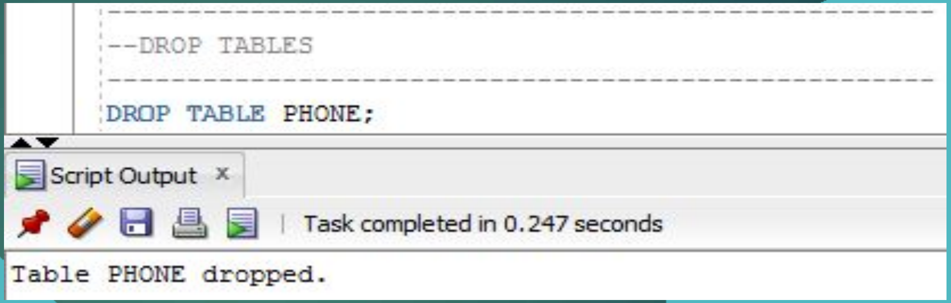
Task completed in 0.177 seconds

```
Table PERSON created.
```

CREATE TABLE phone(
Phone int primary key,
ISSN int,
FOREIGN KEY (ISSN) REFERENCES
Persons(ISSN)
);

SQL: DDL

- DROP TABLE:

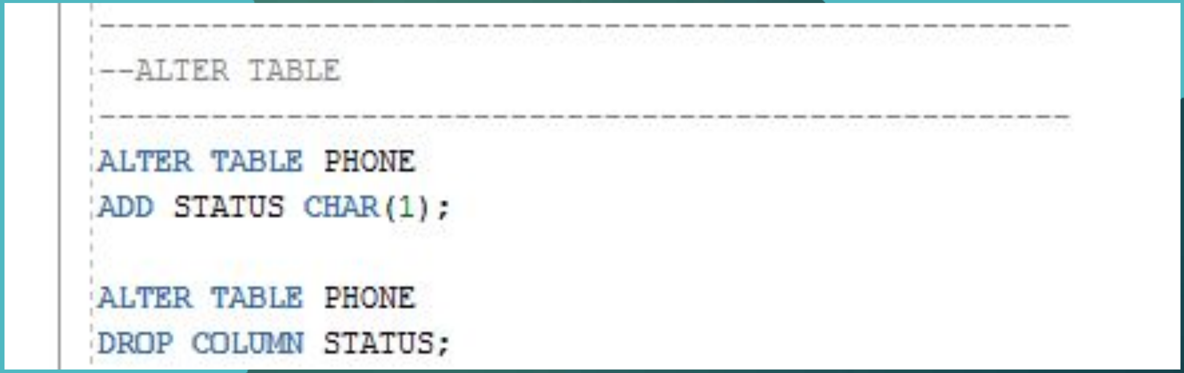


```
--------------------------------------------------------------
--DROP TABLES
--------------------------------------------------------------
DROP TABLE PHONE;
```

```
Task completed in 0.247 seconds
```

```
Table PHONE dropped.
```

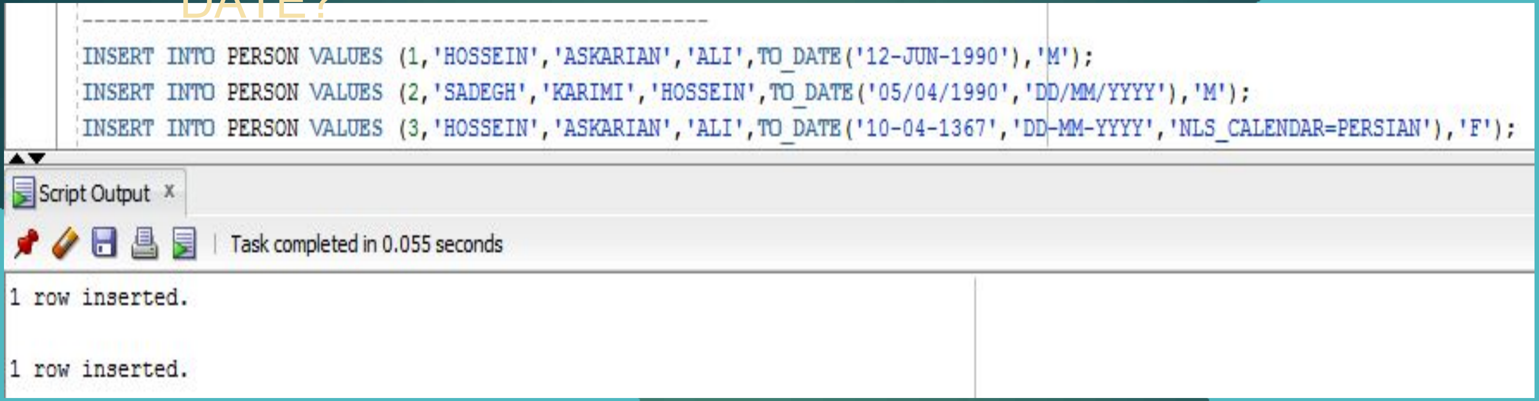
- ALTER TABLE



```
--------------------------------------------------------------
--ALTER TABLE
--------------------------------------------------------------
ALTER TABLE PHONE
ADD STATUS CHAR(1);

ALTER TABLE PHONE
DROP COLUMN STATUS;
```

SQL: DML: INSERT

- INSERT INTO:
  - NUMBER, CHAR/VARCHAR2, DATE?

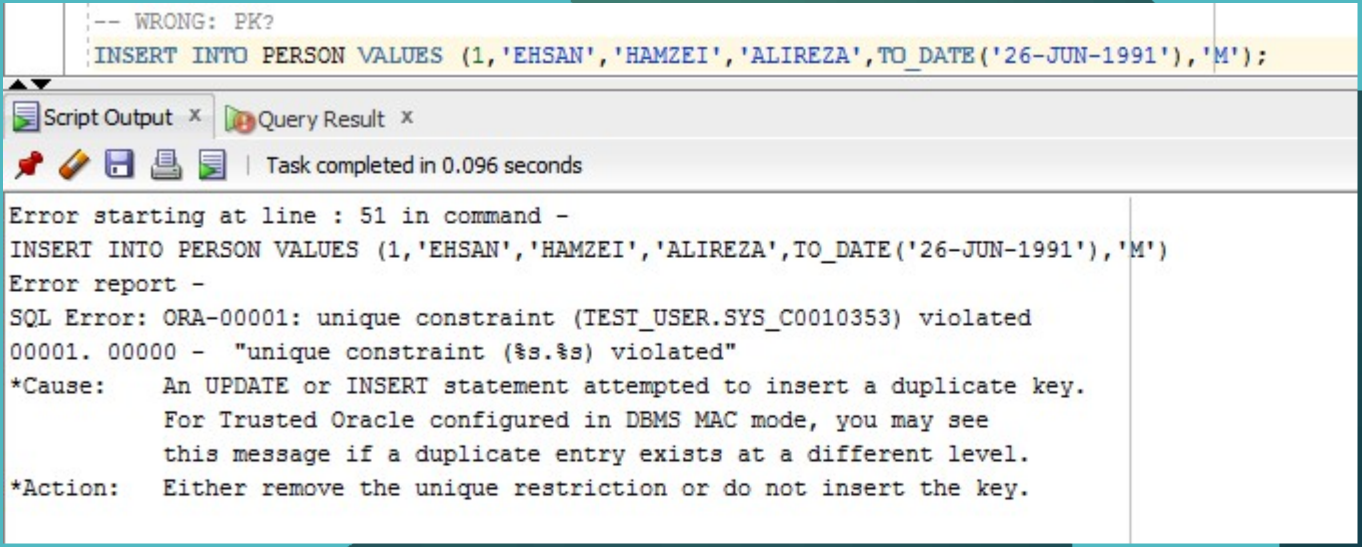


```
INSERT INTO PERSON VALUES (1,'HOSSEIN','ASKARIAN','ALI',TO_DATE('12-JUN-1990'),'M');
INSERT INTO PERSON VALUES (2,'SADEGH','KARIMI','HOSSEIN',TO_DATE('05/04/1990','DD/MM/YYYY'),'M');
INSERT INTO PERSON VALUES (3,'HOSSEIN','ASKARIAN','ALI',TO_DATE('10-04-1367','DD-MM-YYYY','NLS_CALENDAR=PERSIAN'),'F');
```

Script Output

Task completed in 0.055 seconds

```
1 row inserted.

1 row inserted.
```



```
-- WRONG: PK?
INSERT INTO PERSON VALUES (1,'EHSAN','HAMZEI','ALIREZA',TO_DATE('26-JUN-1991'),'M');
```

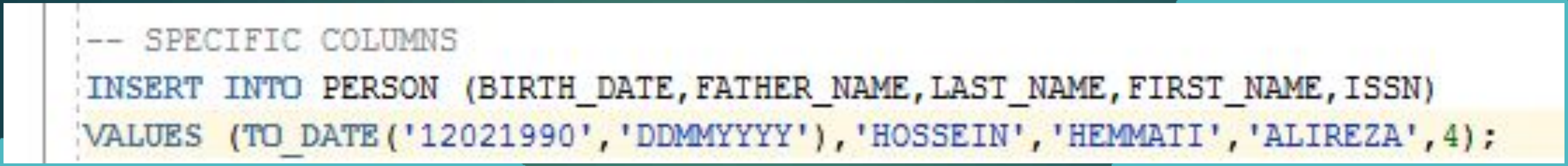
Script Output | Query Result

Task completed in 0.096 seconds

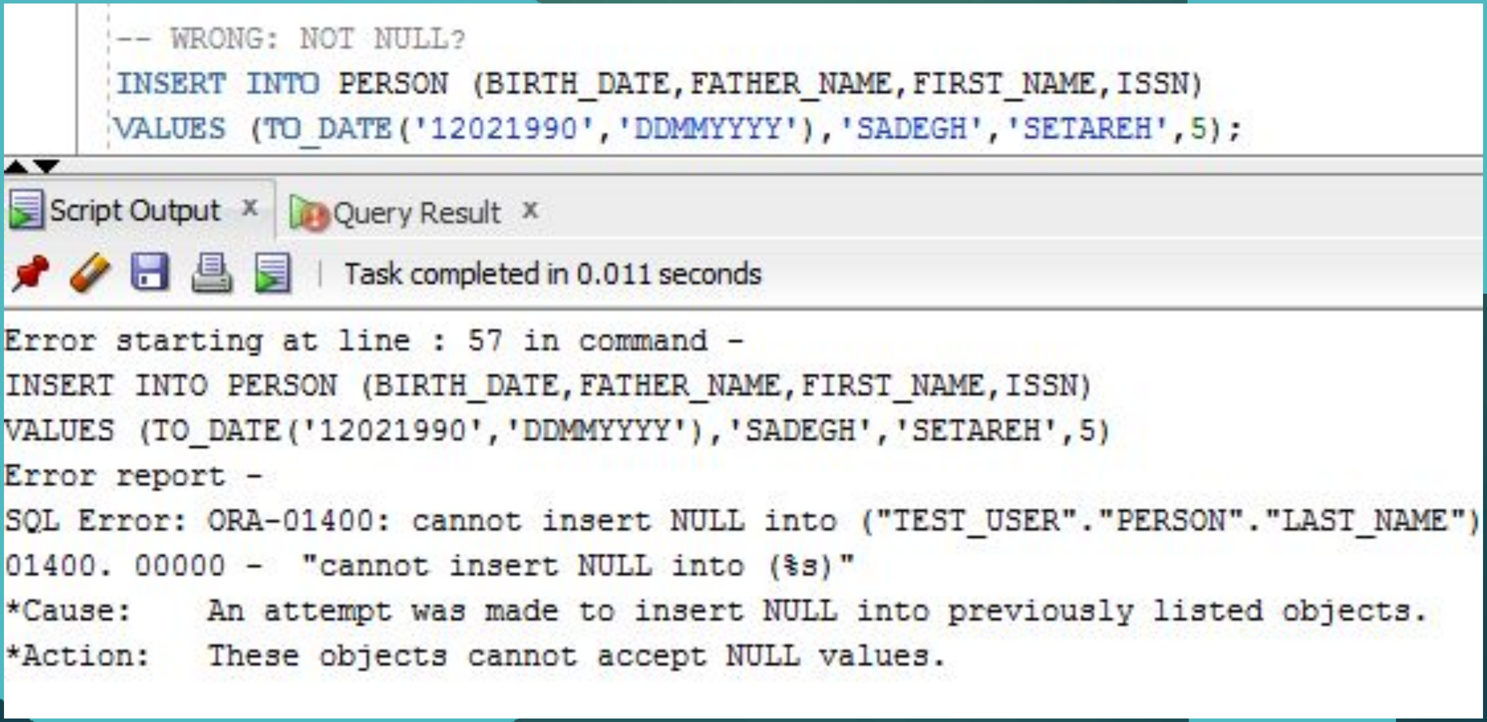
```
Error starting at line : 51 in command -
INSERT INTO PERSON VALUES (1,'EHSAN','HAMZEI','ALIREZA',TO_DATE('26-JUN-1991'),'M')
Error report -
SQL Error: ORA-00001: unique constraint (TEST_USER.SYS_C0010353) violated
00001. 00000 -  "unique constraint (%s.%s) violated"
*Cause:    An UPDATE or INSERT statement attempted to insert a duplicate key.
           For Trusted Oracle configured in DBMS MAC mode, you may see
           this message if a duplicate entry exists at a different level.
*Action:   Either remove the unique restriction or do not insert the key.
```

SQL: DML: INSERT

- INSERT INTO:
  - SPECIFIC COLUMNS



```
-- SPECIFIC COLUMNS
INSERT INTO PERSON (BIRTH_DATE,FATHER_NAME,LAST_NAME,FIRST_NAME,ISSN)
VALUES (TO_DATE('12021990','DDMMYYYY'),'HOSSEIN','HEMMATI','ALIREZA',4);
```

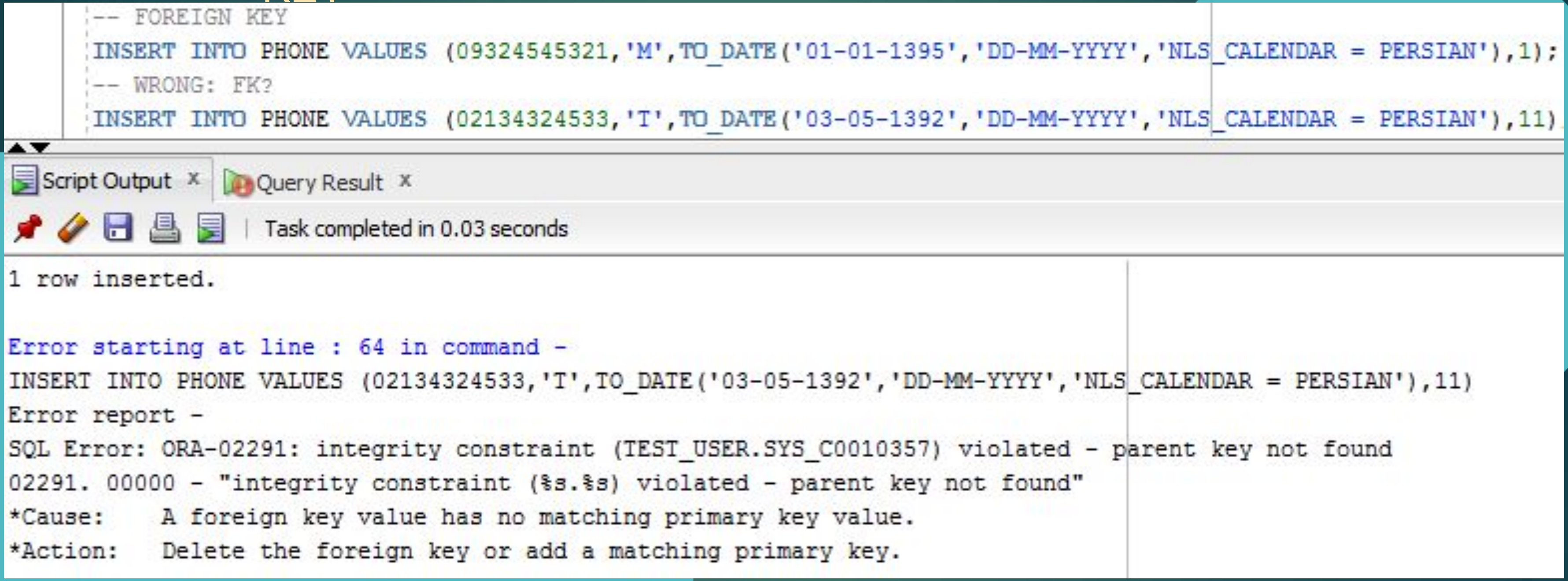


```
-- WRONG: NOT NULL?
INSERT INTO PERSON (BIRTH_DATE,FATHER_NAME,FIRST_NAME,ISSN)
VALUES (TO_DATE('12021990','DDMMYYYY'),'SADEGH','SETAREH',5);
```

```
Error starting at line : 57 in command -
INSERT INTO PERSON (BIRTH_DATE,FATHER_NAME,FIRST_NAME,ISSN)
VALUES (TO_DATE('12021990','DDMMYYYY'),'SADEGH','SETAREH',5)
Error report -
SQL Error: ORA-01400: cannot insert NULL into ("TEST_USER"."PERSON"."LAST_NAME")
01400. 00000 -  "cannot insert NULL into (%s)"
*Cause:    An attempt was made to insert NULL into previously listed objects.
*Action:   These objects cannot accept NULL values.
```

SQL: DML: INSERT

- INSERT INTO:
  - FOREIGN KEY



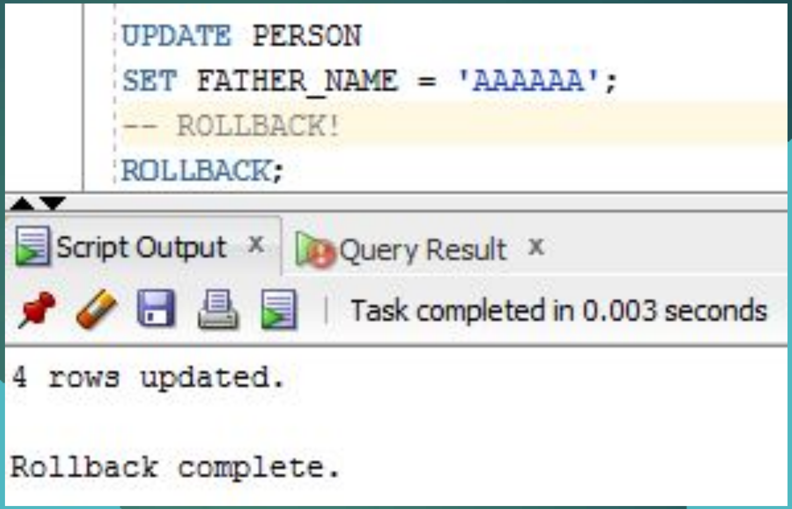
```
-- FOREIGN KEY
INSERT INTO PHONE VALUES (09324545321,'M',TO_DATE('01-01-1395','DD-MM-YYYY','NLS_CALENDAR = PERSIAN'),1);
-- WRONG: FK?
INSERT INTO PHONE VALUES (02134324533,'T',TO_DATE('03-05-1392','DD-MM-YYYY','NLS_CALENDAR = PERSIAN'),11)
```

```
1 row inserted.

Error starting at line : 64 in command -
INSERT INTO PHONE VALUES (02134324533,'T',TO_DATE('03-05-1392','DD-MM-YYYY','NLS_CALENDAR = PERSIAN'),11)
Error report -
SQL Error: ORA-02291: integrity constraint (TEST_USER.SYS_C0010357) violated - parent key not found
02291. 00000 - "integrity constraint (%s.%s) violated - parent key not found"
*Cause:    A foreign key value has no matching primary key value.
*Action:   Delete the foreign key or add a matching primary key.
```

- ALL DML COMMANDS NEED COMMIT

SQL: DML: UPDATE

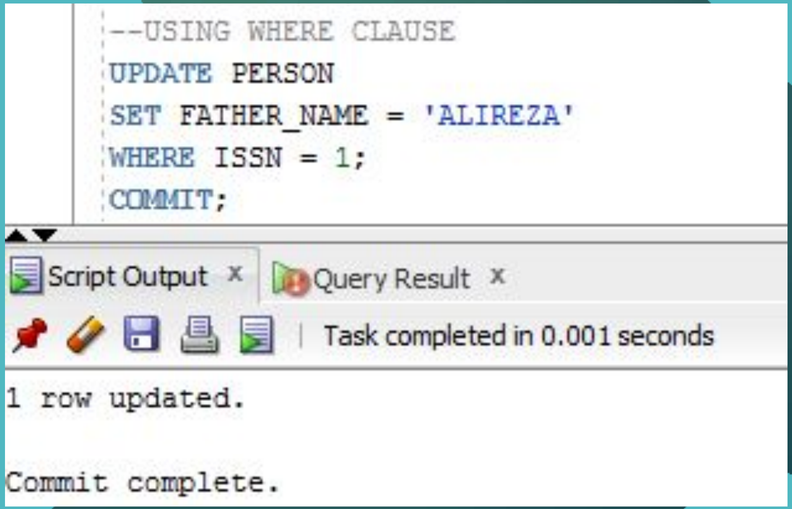
- UPDAT
  E



```
UPDATE PERSON
SET FATHER_NAME = 'AAAAAA';
-- ROLLBACK!
ROLLBACK;
```

```
4 rows updated.

Rollback complete.
```

- WHERE
  CLAUSE

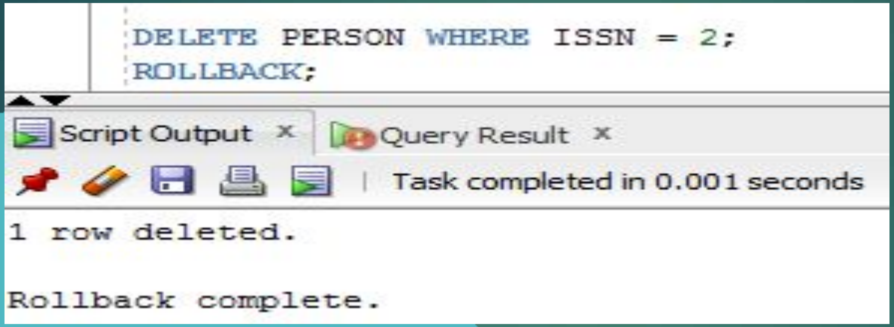


```
--USING WHERE CLAUSE
UPDATE PERSON
SET FATHER_NAME = 'ALIREZA'
WHERE ISSN = 1;
COMMIT;
```

```
1 row updated.

Commit complete.
```

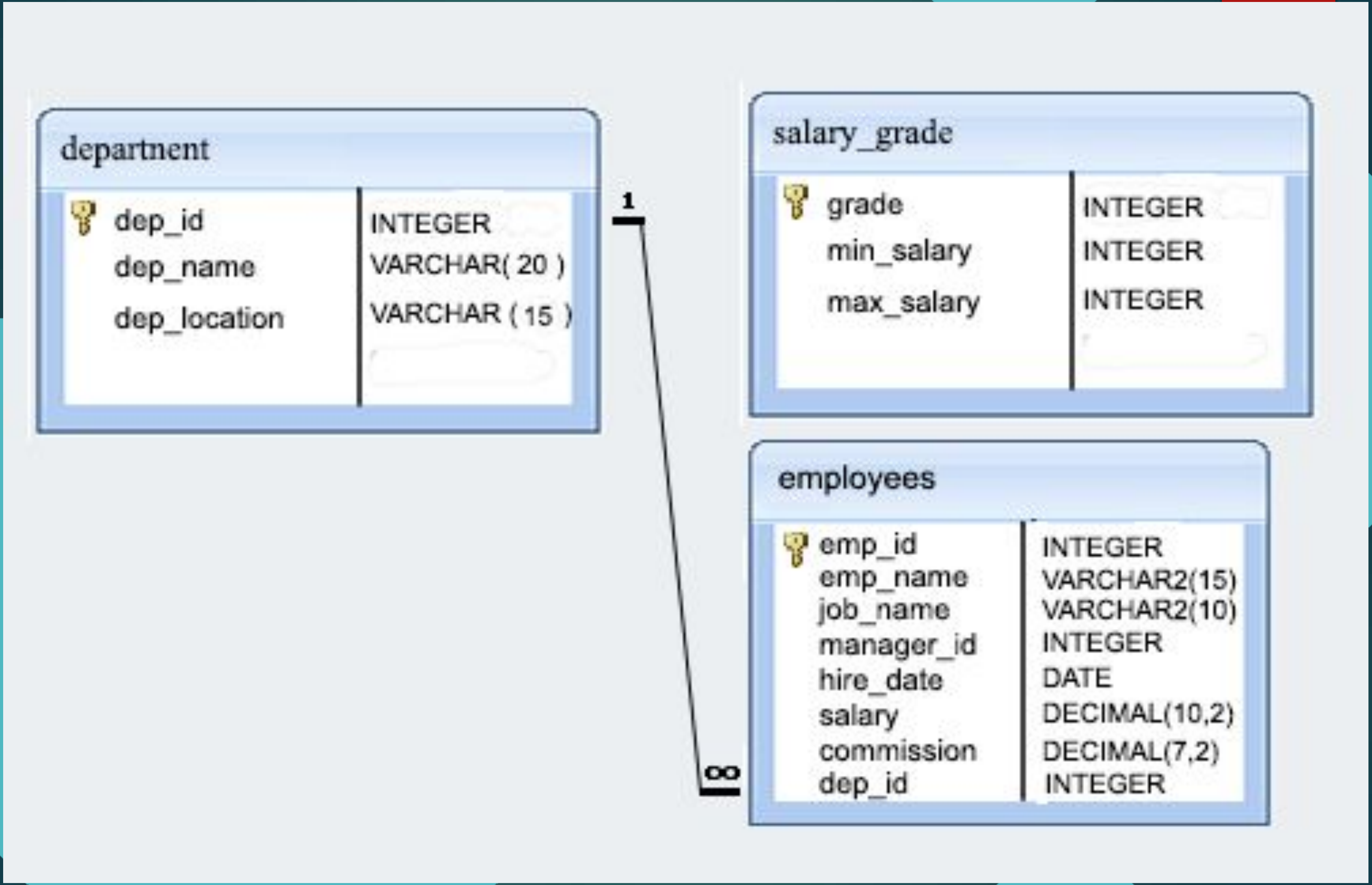
SQL: DML: DELETE

- DELETE



```
DELETE PERSON WHERE ISSN = 2;
ROLLBACK;
```

```
1 row deleted.

Rollback complete.
```

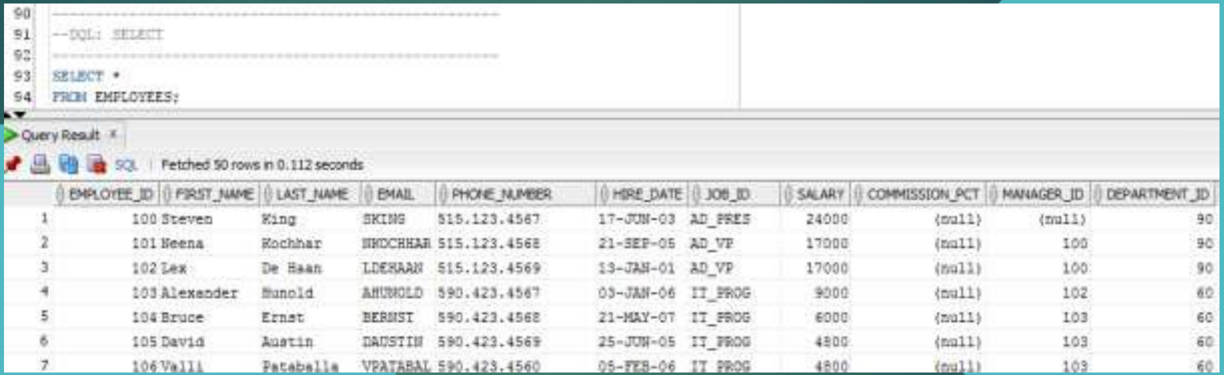


department
dep_id INTEGER
dep_name VARCHAR(20)
dep_location VARCHAR (15)
salary_grade
grade INTEGER
min_salary INTEGER
max_salary INTEGER
employees
emp_id INTEGER
emp_name VARCHAR2(15)
job_name VARCHAR2(10)
manager_id INTEGER
hire_date DATE
salary DECIMAL(10,2)
commission DECIMAL(7,2)
dep_id INTEGER
1
∞

SQL: DQL: SELECT

- SELECT
  - *
  - SPECIFIC COLUMNS

```
SELECT column1, column2....columnN
FROM   table_name;
```

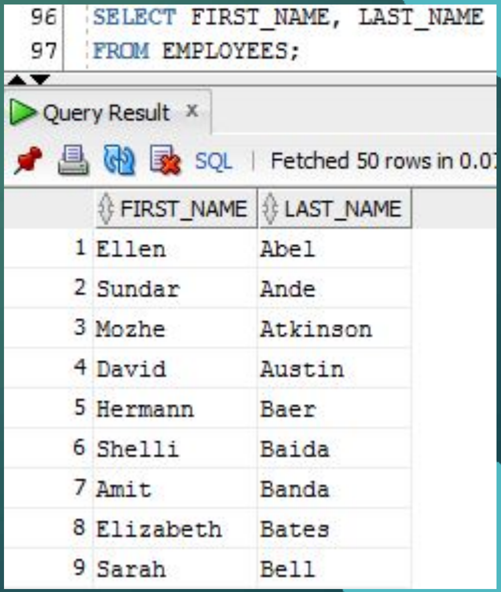


```
------------------------------------------------
--DQL: SELECT
------------------------------------------------
SELECT *
FROM EMPLOYEES;
```

Query Result ×

SQL | Fetched 50 rows in 0.112 seconds

```
  EMPLOYEE_ID FIRST_NAME LAST_NAME EMAIL    PHONE_NUMBER HIRE_DATE JOB_ID  SALARY COMMISSION_PCT MANAGER_ID DEPARTMENT_ID
1         100 Steven     King      SKING    515.123.4567 17-JUN-03 AD_PRES  24000         (null)     (null)            90
2         101 Neena      Kochhar   NKOCHHAR 515.123.4568 21-SEP-05 AD_VP    17000         (null)        100            90
3         102 Lex        De Haan   LDEHAAN  515.123.4569 13-JAN-01 AD_VP    17000         (null)        100            90
4         103 Alexander  Hunold    AHUNOLD  590.423.4567 03-JAN-06 IT_PROG   9000         (null)        102            60
5         104 Bruce      Ernst     BERNST   590.423.4568 21-MAY-07 IT_PROG   6000         (null)        103            60
6         105 David      Austin    DAUSTIN  590.423.4569 25-JUN-05 IT_PROG   4800         (null)        103            60
7         106 Valli      Pataballa VPATABAL 590.423.4560 05-FEB-06 IT_PROG   4800         (null)        103            60
```



```
SELECT FIRST_NAME, LAST_NAME
FROM EMPLOYEES;
```

Query Result ×

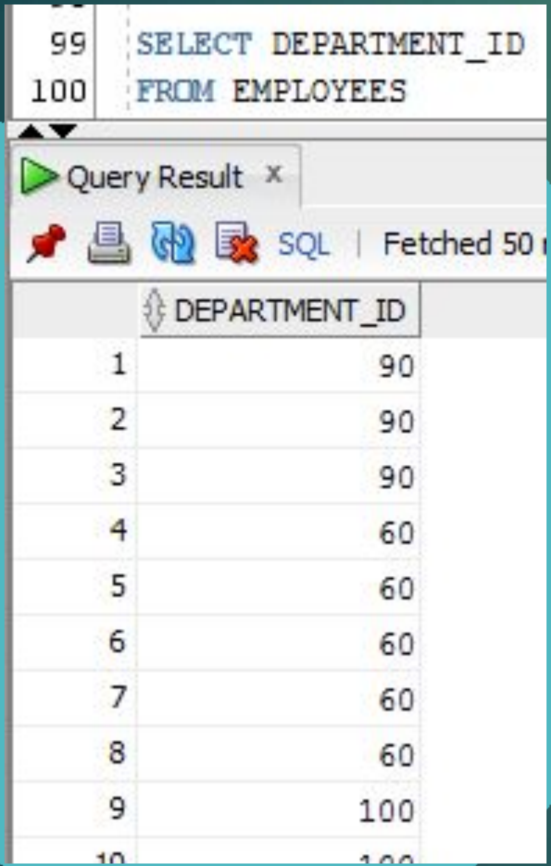
SQL | Fetched 50 rows in 0.0

```
  FIRST_NAME LAST_NAME
1 Ellen      Abel
2 Sundar     Ande
3 Mozhe      Atkinson
4 David      Austin
5 Hermann    Baer
6 Shelli     Baida
7 Amit       Banda
8 Elizabeth  Bates
9 Sarah      Bell
```

SQL: DQL: SELECT

- DISTINCT CLUASE

```
SELECT DISTINCT column1, column2....columnN
FROM    table_name;
```

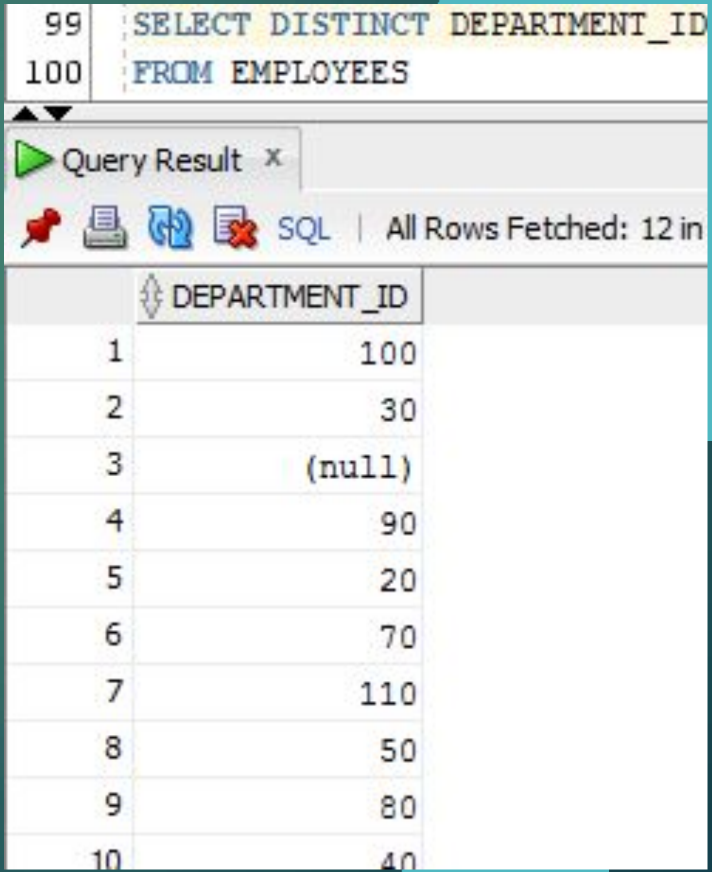


```
SELECT DEPARTMENT_ID
FROM EMPLOYEES
```

Query Result ×

SQL | Fetched 50

	DEPARTMENT_ID
1	90
2	90
3	90
4	60
5	60
6	60
7	60
8	60
9	100



```
SELECT DISTINCT DEPARTMENT_ID
FROM EMPLOYEES
```

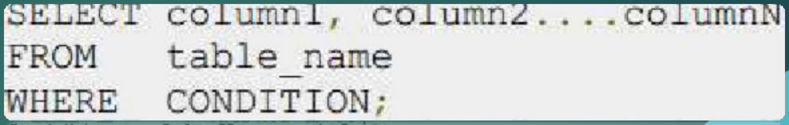
Query Result ×

SQL | All Rows Fetched: 12 in

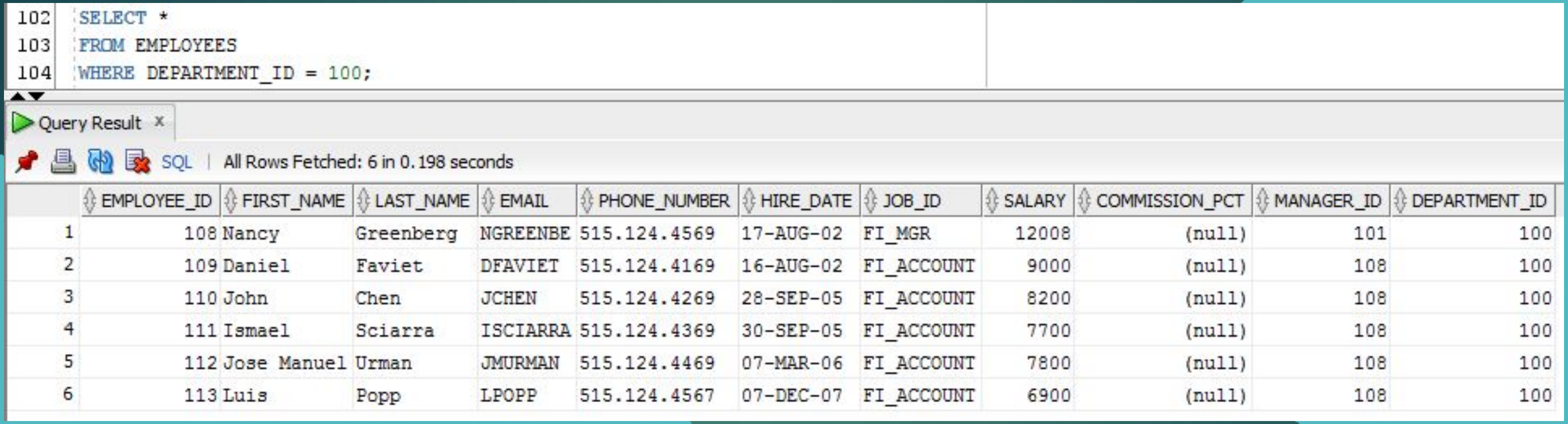
	DEPARTMENT_ID
1	100
2	30
3	(null)
4	90
5	20
6	70
7	110
8	50
9	80
10	40

SQL: DQL: SELECT

- WHERE CLUASE



```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION;
```

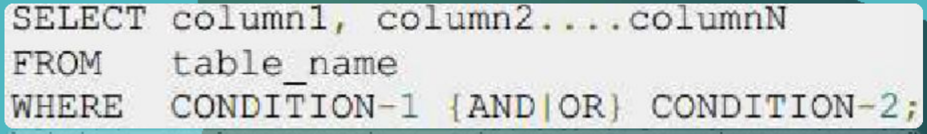


```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID = 100;
```

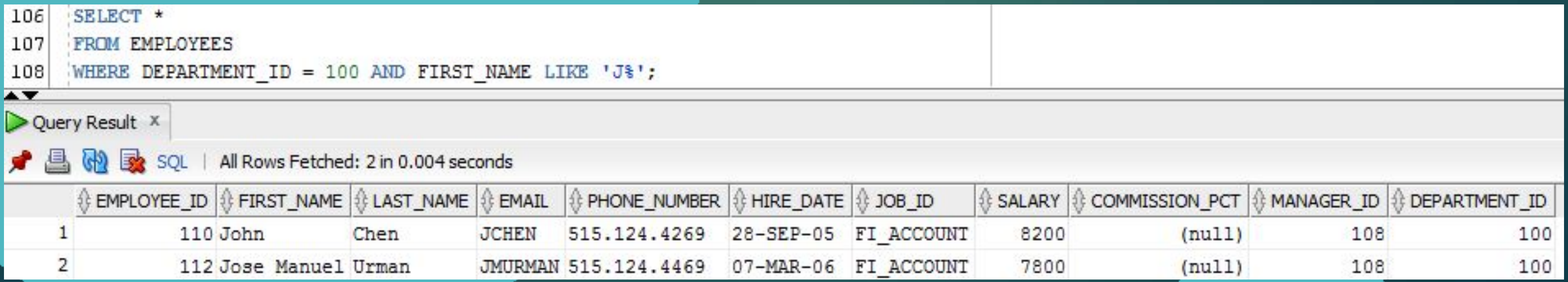
Query Result ×

SQL | All Rows Fetched: 6 in 0.198 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI_MGR	12008	(null)	101	100
2	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000	(null)	108	100
3	110	John	Chen	JCHEN	515.124.4269	28-SEP-05	FI_ACCOUNT	8200	(null)	108	100
4	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI_ACCOUNT	7700	(null)	108	100
5	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI_ACCOUNT	7800	(null)	108	100
6	113	Luis	Popp	LPOPP	515.124.4567	07-DEC-07	FI_ACCOUNT	6900	(null)	108	100



```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION-1 {AND|OR} CONDITION-2;
```



```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID = 100 AND FIRST_NAME LIKE 'J%';
```

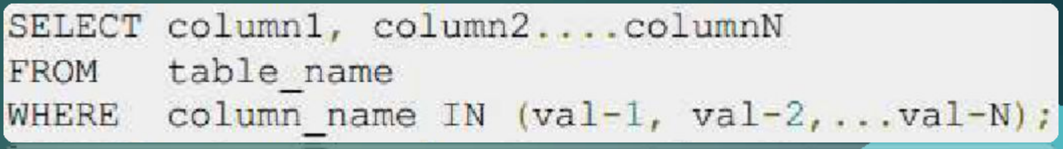
Query Result ×

SQL | All Rows Fetched: 2 in 0.004 seconds

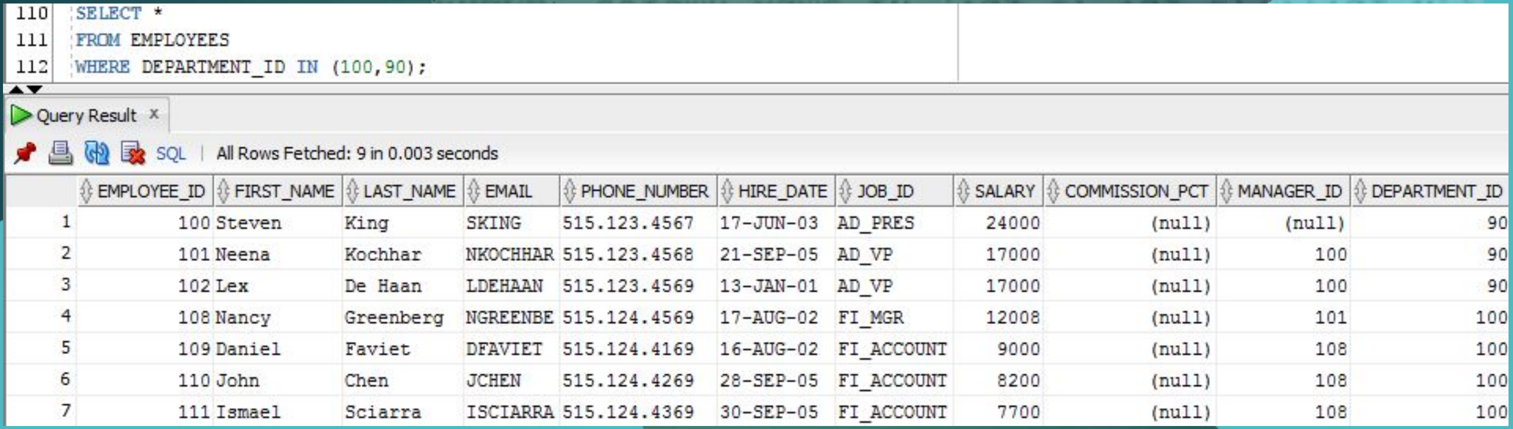
	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	110	John	Chen	JCHEN	515.124.4269	28-SEP-05	FI_ACCOUNT	8200	(null)	108	100
2	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI_ACCOUNT	7800	(null)	108	100

SQL: DQL: SELECT

► IN CLUASE



```
SELECT column1, column2....columnN
FROM   table_name
WHERE  column_name IN (val-1, val-2,...val-N);
```



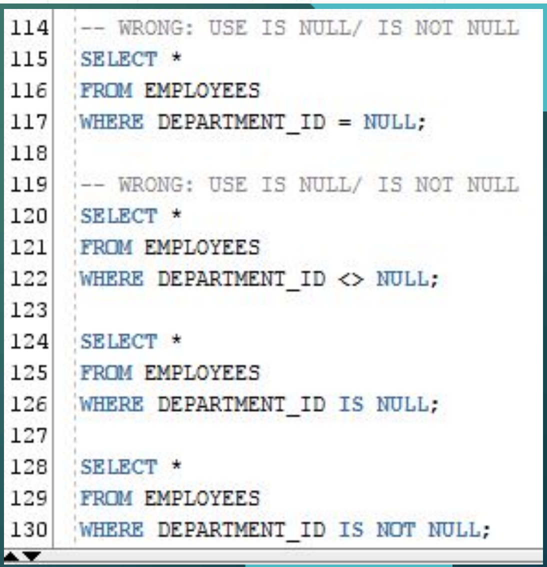
```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID IN (100,90);
```

Query Result ×

SQL | All Rows Fetched: 9 in 0.003 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	(null)	(null)	90
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	(null)	100	90
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	(null)	100	90
4	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI_MGR	12008	(null)	101	100
5	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000	(null)	108	100
6	110	John	Chen	JCHEN	515.124.4269	28-SEP-05	FI_ACCOUNT	8200	(null)	108	100
7	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI_ACCOUNT	7700	(null)	108	100

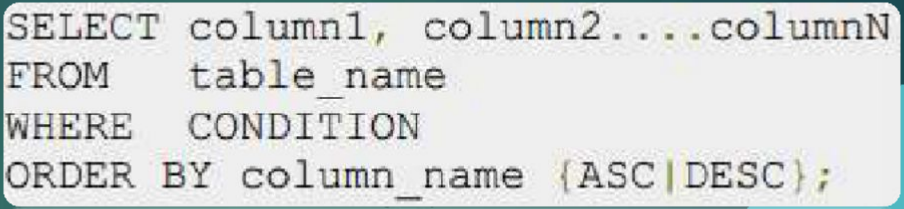
- NULL IN WHERE?
  - IS NULL
  - IS NOT NULL



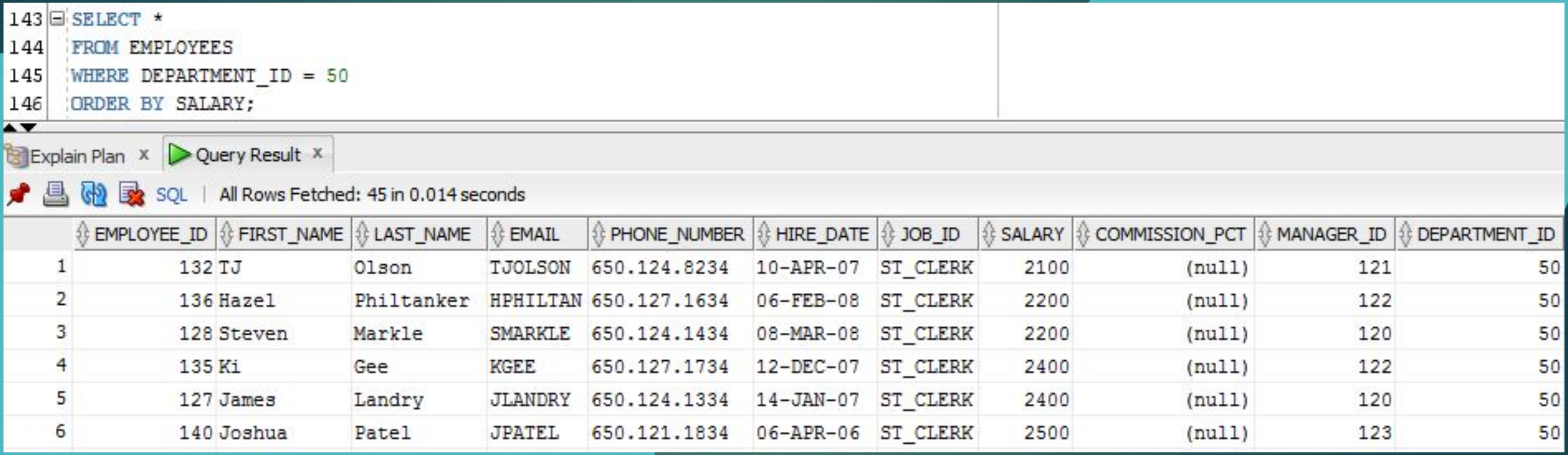
```
-- WRONG: USE IS NULL/ IS NOT NULL
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID = NULL;

-- WRONG: USE IS NULL/ IS NOT NULL
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID <> NULL;

SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID IS NULL;

SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID IS NOT NULL;
```

SQL: DQL: SELECT

- ORDER BY



```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION
ORDER BY column_name {ASC|DESC};
```

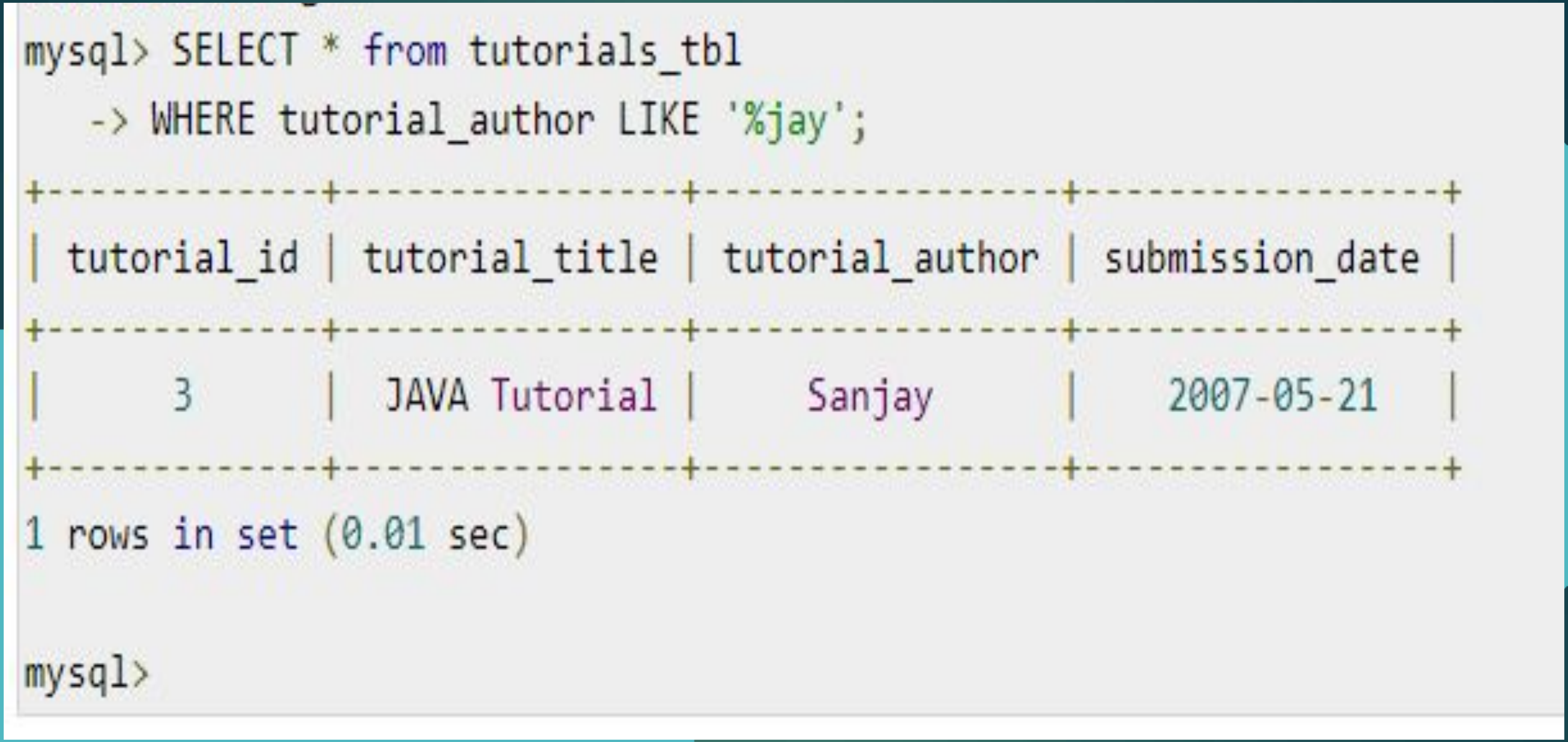


```
SELECT *
FROM EMPLOYEES
WHERE DEPARTMENT_ID = 50
ORDER BY SALARY;
```

All Rows Fetched: 45 in 0.014 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	132	TJ	Olson	TJOLSON	650.124.8234	10-APR-07	ST_CLERK	2100	(null)	121	50
2	136	Hazel	Philtanker	HPHILTAN	650.127.1634	06-FEB-08	ST_CLERK	2200	(null)	122	50
3	128	Steven	Markle	SMARKLE	650.124.1434	08-MAR-08	ST_CLERK	2200	(null)	120	50
4	135	Ki	Gee	KGEE	650.127.1734	12-DEC-07	ST_CLERK	2400	(null)	122	50
5	127	James	Landry	JLANDRY	650.124.1334	14-JAN-07	ST_CLERK	2400	(null)	120	50
6	140	Joshua	Patel	JPATEL	650.121.1834	06-APR-06	ST_CLERK	2500	(null)	123	50

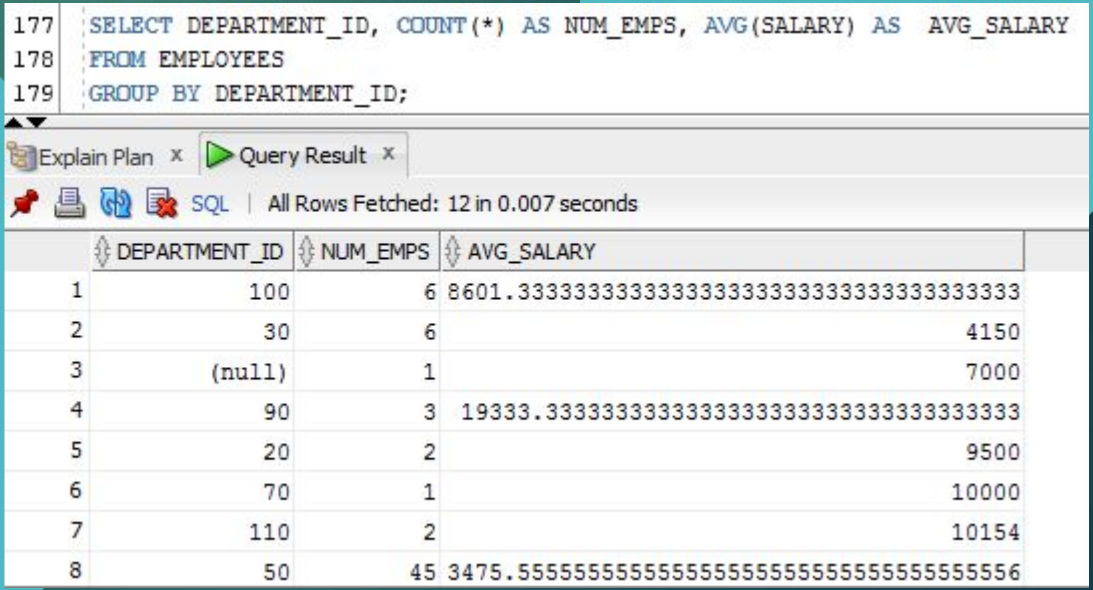
LIKE Clause



```
mysql> SELECT * from tutorials_tbl
   -> WHERE tutorial_author LIKE '%jay';
+-------------+----------------+-----------------+-----------------+
| tutorial_id | tutorial_title | tutorial_author | submission_date |
+-------------+----------------+-----------------+-----------------+
|      3      |  JAVA Tutorial |      Sanjay     |    2007-05-21   |
+-------------+----------------+-----------------+-----------------+
1 rows in set (0.01 sec)

mysql>
```

SQL: DQL: SELECT

- GROUP BY
- AGGREGATE FUCTION
  - MAX,MIN
  - AVG
  - COUNT
  - SUM



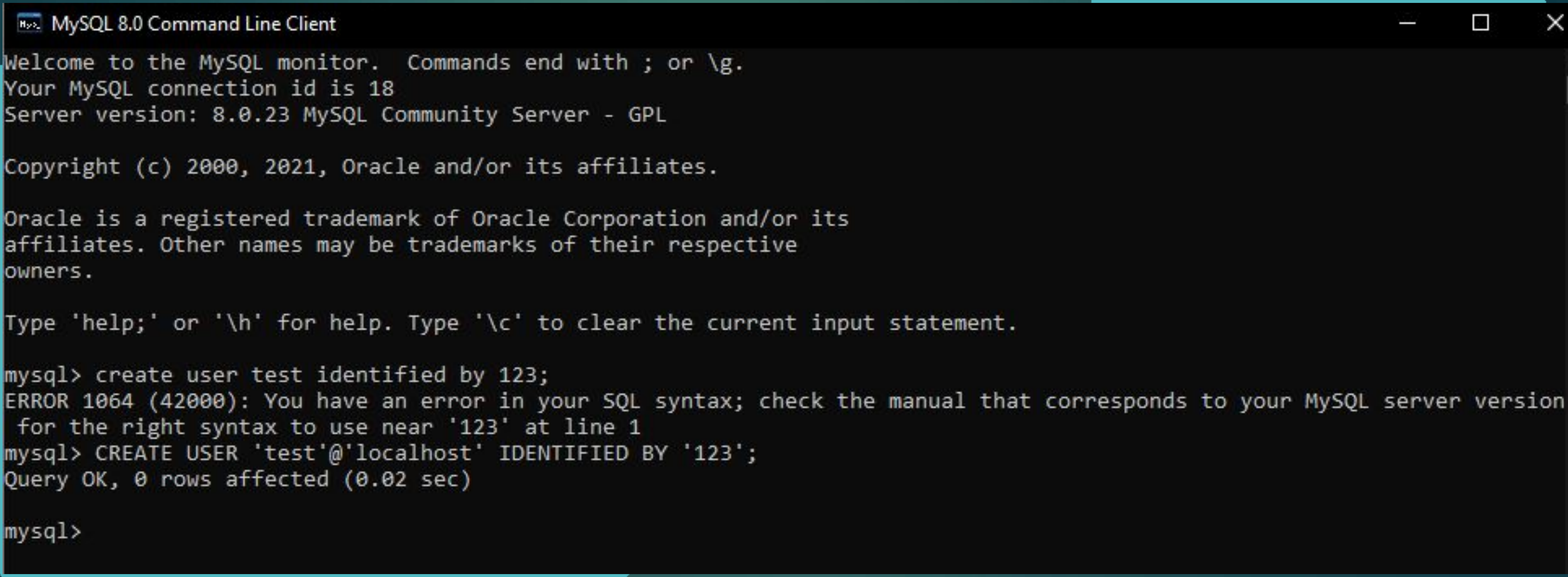
```
SELECT DEPARTMENT_ID, COUNT(*) AS NUM_EMPS, AVG(SALARY) AS  AVG_SALARY
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID;
```

All Rows Fetched: 12 in 0.007 seconds

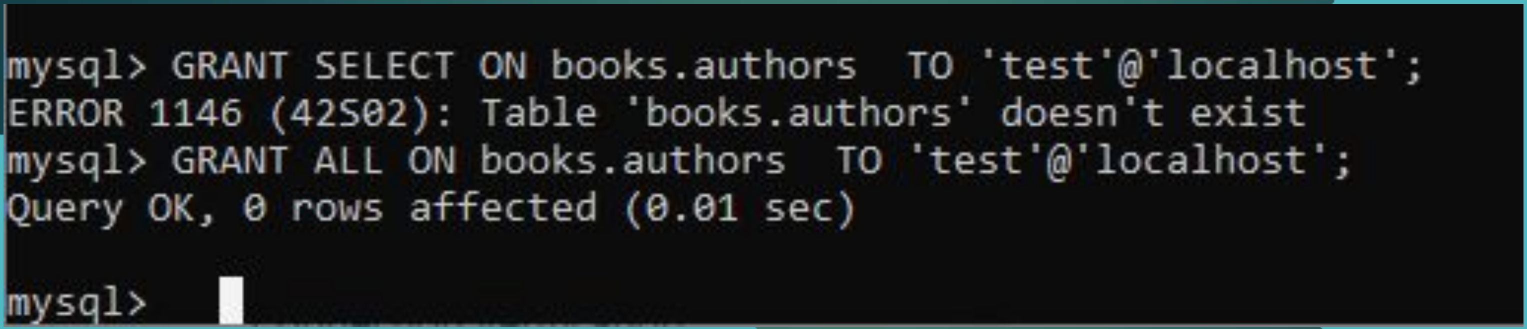
	DEPARTMENT_ID	NUM_EMPS	AVG_SALARY
1	100	6	8601.3333333333333333333333333333333333
2	30	6	4150
3	(null)	1	7000
4	90	3	19333.333333333333333333333333333333333
5	20	2	9500
6	70	1	10000
7	110	2	10154
8	50	45	3475.5555555555555555555555555555555556

DCL: CREATE USER

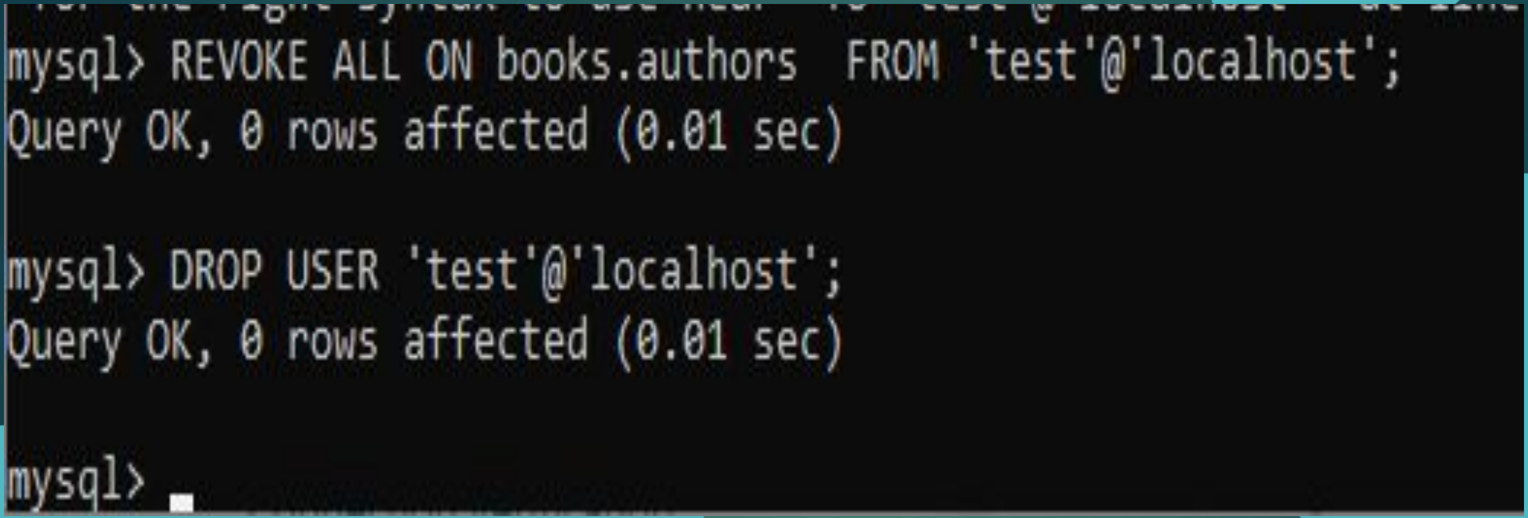
- 1) Define User with Username/Password
- 2) Grants Sufficient Privileges.



```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18
Server version: 8.0.23 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user test identified by 123;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
 for the right syntax to use near '123' at line 1
mysql> CREATE USER 'test'@'localhost' IDENTIFIED BY '123';
Query OK, 0 rows affected (0.02 sec)

mysql>
```



```
mysql> GRANT SELECT ON books.authors  TO 'test'@'localhost';
ERROR 1146 (42S02): Table 'books.authors' doesn't exist
mysql> GRANT ALL ON books.authors  TO 'test'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql>
```



```
for the right syntax to use near 'TO 'test'@'localhost'' at line
mysql> REVOKE ALL ON books.authors  FROM 'test'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> DROP USER 'test'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> _
```

Online Compiler:
https://paiza.io/projects/p1M_17ebkKK5hqnv9-4OZQ?language=mysql

Other database objects
VIEWS
Sequences

Creating Views

Database views are created using the CREATE VIEW statement. Views can be created from a single table, multiple tables or another view.

To create a view, a user must have the appropriate system privilege according to the specific implementation.

The basic CREATE VIEW syntax is as follows −

```
CREATE VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE [condition];
```

Example

Consider the CUSTOMERS table having the following records –

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Following is an example to create a view from the CUSTOMERS table. This view would be used to have customer name and age from the CUSTOMERS table.

```
SQL > CREATE VIEW CUSTOMERS_VIEW AS
SELECT name, age
FROM  CUSTOMERS;
```

Now, you can query CUSTOMERS_VIEW in a similar way as you query an actual table. Following is an example for the same.

SQL > SELECT * FROM CUSTOMERS_VIEW;
This would produce the following result.

```
+----------+-----+
| name     | age |
+----------+-----+
| Ramesh   |  32 |
| Khilan   |  25 |
| kaushik  |  23 |
| Chaitali |  25 |
| Hardik   |  27 |
| Komal    |  22 |
| Muffy    |  24 |
+----------+-----+
```

Dropping Views

Obviously, where you have a view, you need a way to drop the view if it is no longer needed. The syntax is very simple and is given below −

```
DROP VIEW view_name;
```

Following is an example to drop the CUSTOMERS_VIEW from the CUSTOMERS table.

```
DROP VIEW CUSTOMERS_VIEW;
```

MySQL SEQUENCE

A sequence in MySQL is an arrangement of integers generated in the ascending order (1, 2, 3, and so on) on specific demand.
Sequences are used in the databases to generate unique numbers.
Many applications require each row of a table to contain a distinct value, such as student roll number in student_table, employee numbers in HR, customer ID in CRM, etc. To fulfill this type of arrangement, we use sequences that provide an easy way to generate them.

Note :
MySQL does not provide any built-in function to create a sequence for a table's rows or columns. But we can generate it via SQL query. In this article, we are going to describe how to create a sequence in MySQL using SQL query.

The simplest way for creating a sequence in MySQL is by defining the column as **AUTO_INCREMENT** during table creation, which should be a primary key column.

Execute the below query to create a table:

```
mysql> CREATE TABLE Insects (
 Id INT UNSIGNED NOT NULL AUTO_INCREMENT,  PRIMARY KEY (id),  Name VARCHAR(30)
NOT NULL,Type VARCHAR(30) NOT NULL,Origin VARCHAR(30) NOT NULL );
```

```
mysql> INSERT INTO Insects (Name, Type, Origin) VALUES ('Cockroach', 'Crawling',
'Kitchen’);
```

THANK YOU