

# DB-LAB-TASK-2

Roll No: 24P-0706

Name: Aazan Noor Khuwaja

Dept: BS-CS

Section : 4B

## ALTER TABLE PART:

### 1. Add a column education\_level:

#### Command Input:

```
aazan-noor-khuwaja@Hp-G3:~/Aazan_Data/4th_Semester/DB_Lab/lab2$ sqlite3 tech_survey.db
SQLite version 3.45.1 2024-01-30 16:01:20
Enter ".help" for usage hints.
sqlite> ALTER TABLE "developers"
...> ADD education_level TEXT;
sqlite>
```

#### Command:

```
ALTER TABLE "developers"
ADD education_level TEXT;
```

#### Output:

DB Browser for SQLite - /home/aaazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

The screenshot shows the 'developers' table with 20 rows of data. The columns are:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary	education_level
1	Pakistan	2	Python	JavaScript	MySQL	Django	8000	NULL
2	India	5	Java	SQL	Oracle	Spring	18000	NULL
3	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000	NULL
4	Germany	7	C#	SQL	SQL Server	.NET	70000	NULL
5	Pakistan	1	C	NULL	NULL	NULL	3000	NULL
6	UK	12	Python	R	PostgreSQL	Flask	85000	NULL
7	Canada	4	JavaScript	Python	MongoDB	Vue	60000	NULL
8	Pakistan	3	Java	Kotlin	MySQL	Spring	12000	NULL
9	USA	15	C++	Python	SQLite	Qt	110000	NULL
10	India	6	PHP	JavaScript	MySQL	Laravel	15000	NULL
11	France	8	Python	SQL	PostgreSQL	Django	65000	NULL
12	Australia	9	JavaScript	NULL	MongoDB	React	75000	NULL
13	Japan	11	Java	Python	Oracle	Spring	82000	NULL
14	Brazil	4	Python	JavaScript	MySQL	Flask	22000	NULL
15	Netherlands	6	Go	Python	PostgreSQL	NULL	68000	NULL
16	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000	NULL
17	Pakistan	5	C++	Python	MySQL	NULL	14000	NULL
18	Sweden	10	Python	Julia	SQLite	Django	90000	NULL
19	India	2	JavaScript	HTML	Firebase	Angular	10000	NULL
20	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000	NULL

At the bottom right, there is an 'Edit Database Cell' panel with a text input field containing '1'. Below it, a message says 'Type of data currently in cell: Text / Numeric 1 character(s)'. There are tabs for 'Remote', 'Identity' (selected), 'DBHub.io', 'Local', and 'Current Database'. A 'Name' column is listed.

## 2. Rename table developers to dev\_survey:

Command Input:

```
aazan-noor-khuwaja@Hp-G3:~/Aazan_Data/4th_Semester/DB_Lab/lab2$ sqlite3 tech_survey.db
SQLite version 3.45.1 2024-01-30 16:01:20
Enter ".help" for usage hints.
sqlite> ALTER TABLE "developers"
...> ADD education_level TEXT;
sqlite> ALTER TABLE "developers"
...> RENAME TO dev_survey
...> RENAME TO dev_survey;
Parse error: near "RENAME": syntax error
    ALTER TABLE "developers" RENAME TO dev_survey RENAME TO dev_survey;
                           error here ---^
sqlite> RENAME TO dev_survey;
Parse error: near "RENAME": syntax error
    RENAME TO dev_survey;
                           ^--- error here
sqlite> ALTER TABLE "developers"
...> RENAME TO dev_survey;
sqlite>
```

Command:

```
ALTER TABLE "developers"
```

```
RENAME TO dev_survey;
```

## Output:

The screenshot shows two windows of DB Browser for SQLite. The top window displays the database structure for 'tech\_survey.db'. It shows a single table named 'dev\_survey' with the following schema:

```
CREATE TABLE "dev_survey" ( dev_id INTEGER PRIMARY KEY, country TEXT, experience_years INTEGER, primary_language TEXT, secondary_language TEXT, database_used TEXT, framework TEXT, annual_salary INTEGER, education_level TEXT )
```

The bottom window shows the data for the 'dev\_survey' table. The table has 20 rows of data, each representing a developer's survey entry. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, annual\_salary, and education\_level.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary	education_level
1	Pakistan	2	Python	JavaScript	MySQL	Django	8000	NULL
2	India	5	Java	SQL	Oracle	Spring	18000	NULL
3	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000	NULL
4	Germany	7	C#	SQL	SQL Server	.NET	70000	NULL
5	Pakistan	1	C	NULL	NULL	NULL	3000	NULL
6	UK	12	Python	R	PostgreSQL	Flask	85000	NULL
7	Canada	4	JavaScript	Python	MongoDB	Vue	60000	NULL
8	Pakistan	3	Java	Kotlin	MySQL	Spring	12000	NULL
9	USA	15	C++	Python	SQLite	Qt	110000	NULL
10	India	6	PHP	JavaScript	MySQL	Laravel	15000	NULL
11	France	8	Python	SQL	PostgreSQL	Django	65000	NULL
12	Australia	9	JavaScript	NULL	MongoDB	React	75000	NULL
13	Japan	11	Java	Python	Oracle	Spring	82000	NULL
14	Brazil	4	Python	JavaScript	MySQL	Flask	22000	NULL
15	Netherlands	6	Go	Python	PostgreSQL	NULL	68000	NULL
16	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000	NULL
17	Pakistan	5	C++	Python	MySQL	NULL	14000	NULL
18	Sweden	10	Python	Julia	SQLite	Django	90000	NULL
19	India	2	JavaScript	HTML	Firebase	Angular	10000	NULL
20	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000	NULL

### 3. Rename column salary\_usd to annual\_salary:

#### Command Input:

```
aazan-noor-khuwaja@Hp-G3: ~/Aazan_Data/4th_Semester/DB_Lab/lab2
sqlite> ALTER TABLE "dev_survey"
...> RENAME "annual_salary" TO "salary_usd";
sqlite> ALTER TABLE "dev_survey"
...> RENAME "salary_usd" TO "annual_salary";
sqlite>
```

#### Command:

```
ALTER TABLE "dev_survey"
```

RENAME "salary\_usd" TO "annual\_salary";

## Output:

The screenshot shows the DB Browser for SQLite interface with the 'dev\_survey' table selected. The table structure is as follows:

dev_id	country	experience_years	primary_language	secondary_language	database_used	salary_usd
1	Pakistan	1	C	NULL	MySQL	3000
2	Pakistan	2	Python	JavaScript	Django	8000
3	India	2	JavaScript	HTML	Firebase	10000
4	Pakistan	3	Java	Kotlin	MySQL	12000
5	Pakistan	5	C++	Python	MySQL	14000
6	India	6	PHP	JavaScript	MySQL	15000
7	India	5	Java	SQL	Laravel	18000
8	Brazil	4	Python	JavaScript	Oracle	22000
9	South Africa	7	Python	JavaScript	FastAPI	55000
10	Canada	4	JavaScript	Python	MongoDB	60000
11	France	8	Python	SQL	PostgreSQL	65000
12	Netherlands	6	Go	Python	MySQL	68000
13	Germany	7	C#	SQL	SQL Server	70000
14	USA	3	Ruby	JavaScript	MySQL	72000
15	Australia	9	JavaScript	NULL	MongoDB	75000
16	Japan	11	Java	Python	Oracle	82000
17	UK	6	Python	R	PostgreSQL	85000
18	USA	10	TypeScript	JavaScript	React	90000
19	Sweden	10	Python	Julia	SQLite	90000
20	USA	9	C++	Python	Qt	110000

Figure 1 before

The screenshot shows the DB Browser for SQLite interface with the 'dev\_survey' table selected. The table structure is as follows:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary	education_level
1	Pakistan	1	C	NULL	MySQL	3000	NULL	
2	Pakistan	2	Python	JavaScript	MySQL	Django	8000	NULL
3	India	2	JavaScript	HTML	Firebase	Angular	10000	NULL
4	Pakistan	3	Java	Kotlin	MySQL	Spring	12000	NULL
5	Pakistan	5	C++	Python	MySQL	MySQL	14000	NULL
6	India	6	PHP	JavaScript	MySQL	Laravel	15000	NULL
7	India	5	Java	SQL	Oracle	Spring	18000	NULL
8	Brazil	4	Python	JavaScript	MySQL	Flask	22000	NULL
9	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000	NULL
10	Canada	4	JavaScript	Python	MongoDB	Vue	60000	NULL
11	France	8	Python	SQL	PostgreSQL	Django	65000	NULL
12	Netherlands	6	Go	Python	PostgreSQL	MySQL	68000	NULL
13	Germany	7	C#	SQL	SQL Server	.NET	70000	NULL
14	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000	NULL
15	Australia	9	JavaScript	NULL	MongoDB	React	75000	NULL
16	Japan	11	Java	Python	Oracle	Spring	82000	NULL
17	UK	6	Python	R	PostgreSQL	Flask	85000	NULL
18	USA	10	TypeScript	JavaScript	PostgreSQL	React	90000	NULL
19	Sweden	10	Python	Julia	SQLite	Django	90000	NULL
20	USA	9	C++	Python	SQLite	Qt	110000	NULL

Figure 2 After

#### 4. Drop column education\_level:

Command Input:

```
aazan-noor-khuwaja@Hp-G3: ~/Aazan_Data/4th_Semester/DB_Lab/lab2
sqlite> ALTER TABLE "dev_survey"
...> DROP COLUMN "education_level";
Parse error: near ""education_level"": syntax error
ALTER TABLE "dev_survey" DROP COLUMN "education_level";
          error here ...^
sqlite> ALTER TABLE "dev_survey"      []
...> DROP COLUMN "education_level";
sqlite>
```

Command:

```
ALTER TABLE "dev_survey"
DROP COLUMN "education_level";
```

Output:

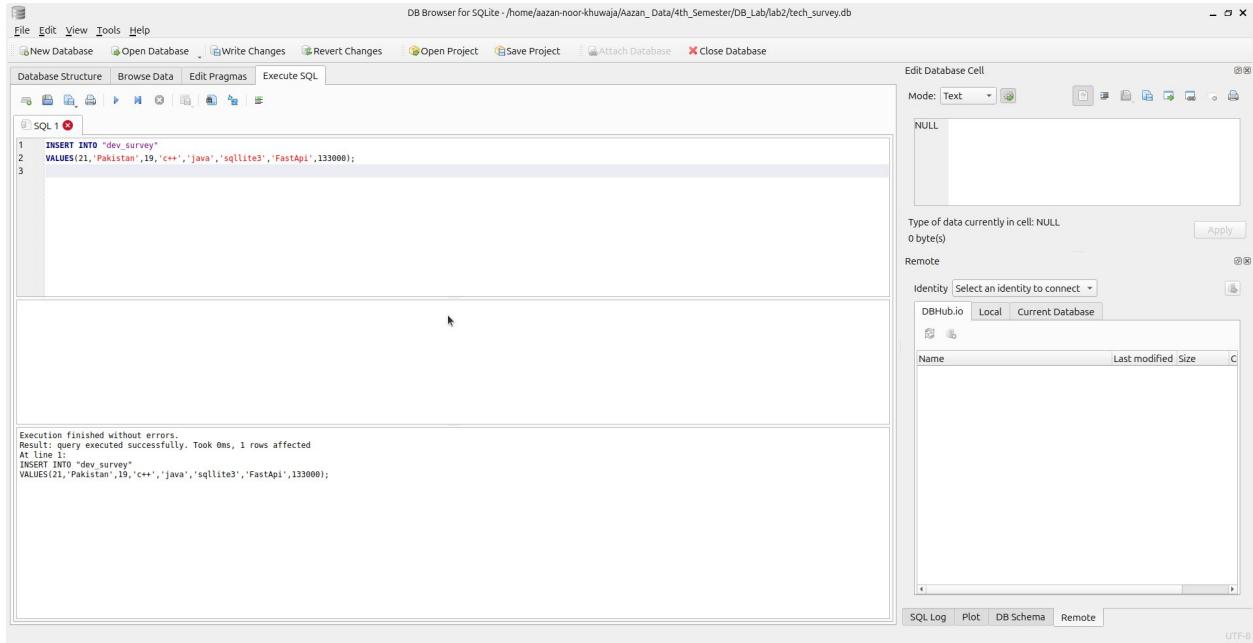
The screenshot shows the DB Browser for SQLite interface. The title bar reads "DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db". The main window displays the "dev\_survey" table with the following columns: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data consists of 20 rows of developer information. To the right of the table, there is an "Edit Database Cell" panel showing the value "1" in a cell, with options for Mode (Text), Apply, and Remote. Below the table, there are navigation buttons for page 1 of 20, a Go to input field, and tabs for SQL Log, Plot, DB Schema, and Remote. The status bar at the bottom right indicates UTF-8 encoding.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	5 Pakistan	1 C	NULL	NULL	MySQL	Django	3000
2	1 Pakistan	2 Python	JavaScript	HTML	Firebase	Angular	8000
3	19 India	2 JavaScript	HTML	MySQL	Spring	10000	
4	8 Pakistan	3 Java	Kotlin	MySQL	Spring	12000	
5	17 Pakistan	5 C++	Python	MySQL	MySQL	14000	
6	10 India	6 PHP	JavaScript	MySQL	Laravel	15000	
7	2 India	5 Java	SQL	Oracle	Spring	18000	
8	14 Brazil	4 Python	JavaScript	MySQL	Flask	22000	
9	20 South Africa	7 Python	JavaScript	PostgreSQL	FastAPI	55000	
10	7 Canada	4 JavaScript	Python	MongoDB	Vue	60000	
11	11 France	8 Python	SQL	PostgreSQL	Django	65000	
12	15 Netherlands	6 Go	Python	PostgreSQL	MySQL	68000	
13	4 Germany	7 C#	SQL	SQL Server	.NET	70000	
14	16 USA	3 Ruby	JavaScript	PostgreSQL	Rails	72000	
15	12 Australia	9 JavaScript	NULL	MongoDB	React	75000	
16	13 Japan	11 Java	Python	Oracle	Spring	82000	
17	6 UK	12 Python	R	PostgreSQL	Flask	85000	
18	3 USA	10 JavaScript	TypeScript	PostgreSQL	React	90000	
19	18 Sweden	10 Python	Julia	SQLite	Django	90000	
20	9 USA	15 C++	Python	SQLite	Qt	110000	

# INSERT PART:

## 1. Insert all columns for one record:

### Command Input:



The screenshot shows the DB Browser for SQLite interface. In the top-left panel, there is an SQL editor window titled "SQL 1" containing the following code:

```
1 INSERT INTO "dev_survey"
2 VALUES(21,'Pakistan',19,'c++','java','sqlite3','FastApi',133000);
3
```

Below the SQL editor, the status bar displays the execution results:

```
Execution finished without errors.
Result: query executed successfully. Took 0ms, 1 rows affected
At line 1:
INSERT INTO "dev_survey"
VALUES(21,'Pakistan',19,'c++','java','sqlite3','FastApi',133000);
```

The right side of the interface features a "Edit Database Cell" panel with a "Mode: Text" dropdown set to "Text". The cell content is "NULL". Below this, a message states "Type of data currently in cell: NULL 0 byte(s)". There are tabs for "DBHub.io", "Local", and "Current Database". The bottom navigation bar includes "SQL Log", "Plot", "DB Schema", and "Remote".

### Command:

```
INSERT INTO "dev_survey"
VALUES(21,'Pakistan',19,'c++','java','sqlite3','FastApi',133000);
```

### Output:

DB Browser for SQLite - /home/aazan-noor-khwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

The table data is as follows:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	1	C	NULL	NULL	NULL	3000
2	Pakistan	2	Python	JavaScript	MySQL	Django	8000
3	India	2	JavaScript	HTML	Firebase	Angular	10000
4	Pakistan	3	Java	Kotlin	MySQL	Spring	12000
5	Pakistan	5	C++	Python	MySQL	NULL	14000
6	India	6	PHP	JavaScript	MySQL	Laravel	15000
7	India	5	Java	SQL	Oracle	Spring	18000
8	Brazil	4	Python	JavaScript	MySQL	Flask	22000
9	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
10	Canada	4	JavaScript	Python	MongoDB	Vue	60000
11	France	8	Python	SQL	PostgreSQL	Django	65000
12	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
13	Germany	7	C#	SQL	SQL Server	.NET	70000
14	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000
15	Australia	9	JavaScript	NULL	MongoDB	React	75000
16	Japan	11	Java	Python	Oracle	Spring	82000
17	UK	12	Python	R	PostgreSQL	Flask	85000
18	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
19	Sweden	10	Python	Julia	SQLite	Django	90000
20	USA	15	C++	Python	SQLite	Qt	110000
21	Pakistan	19	c++	java	sqlite3	FastAPI	133000

1 row(s), 8 column(s). Sum: 133040; Average: 16630; Min: 0; Max: 133000

## 2. Insert specific columns only for one record:

### Command Input:

DB Browser for SQLite - /home/aazan-noor-khwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

```
SQL 1
1 INSERT INTO "dev_survey"(country,secondary_language,annual_salary)
2 VALUES('Vietnam','PHP',117000);
3
```

Execution finished without errors.  
Result: query executed successfully. Took 0ms, 1 rows affected  
All rows shown

```
INSERT INTO "dev_survey"(country,secondary_language,annual_salary)  
VALUES('Vietnam','PHP',117000);
```

### Command:

```
INSERT INTO "dev_survey"(country,secondary_language,annual_salary)
VALUES('Vietnam','PHP',117000);
```

### Output:

DB Browser for SQLite - /home/aazan-noor-khwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

Table: dev\_survey

	dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	5	Pakistan	1	C	NULL	NULL	NULL	3000
2	1	Pakistan	2	Python	JavaScript	MySQL	Django	8000
3	19	India	2	JavaScript	HTML	Firebase	Angular	10000
4	8	Pakistan	3	Java	Kotlin	MySQL	Spring	12000
5	17	Pakistan	5	C++	Python	MySQL	NULL	14000
6	10	India	6	PHP	JavaScript	MySQL	Laravel	15000
7	2	India	5	Java	SQL	Oracle	Spring	18000
8	14	Brazil	4	Python	JavaScript	MySQL	Flask	22000
9	20	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
10	7	Canada	4	JavaScript	Python	MongoDB	Vue	60000
11	11	France	8	Python	SQL	PostgreSQL	Django	65000
12	15	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
13	4	Germany	7	C#	SQL	SQL Server	.NET	70000
14	16	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000
15	12	Australia	9	JavaScript	NULL	MongoDB	React	75000
16	13	Japan	11	Java	Python	Oracle	Spring	82000
17	6	UK	12	Python	R	PostgreSQL	Flask	85000
18	3	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
19	18	Sweden	10	Python	Julia	SQLite	Django	90000
20	9	USA	15	C++	Python	SQLite	Qt	110000
21	22	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
22	21	Pakistan	19	c++	java	sqlite3	FastAPI	133000

1 1 - 22 1 Go to: 1 SQL Log Plot DB Schema Remote UTF-8

### 3. Insert 3 rows in one statement:

#### Command Input:

DB Browser for SQLite - /home/aazan-noor-khwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

```
SQL 1
1 INSERT INTO "dev_survey"
2 VALUES
3 (25, 'Pakistan', '19', 'c++', 'java', 'sqlite3', 'FastApi', 133000),
4 (24, 'INDIA', '23', 'c++', 'java', 'sqlite3', 'FastApi', 131000),
5 (26, 'Bangladesh', '7', 'c++', 'java', 'sqlite3', 'FastApi', 132000);
6
7
```

Execution finished without errors.  
Result: query executed successfully. Took 0ms, 3 rows affected  
All rows inserted.

```
INSERT INTO "dev_survey"
VALUES
(25, 'Pakistan', '19', 'c++', 'java', 'sqlite3', 'FastApi', 133000),
(24, 'INDIA', '23', 'c++', 'java', 'sqlite3', 'FastApi', 131000),
(26, 'Bangladesh', '7', 'c++', 'java', 'sqlite3', 'FastApi', 132000);
```

SQL Log Plot DB Schema Remote UTF-8

#### Command:

```
INSERT INTO "dev_survey"
```

```
VALUES
```

```
(25,'Pakistan',19,'c++','java','sqlite3','FastApi',133000),
```

```
(24,'INDia',23,'c++','java','sqllite3','FastApi',131000),
(26,'Bangladesh',7,'c++','java','sqllite3','FastApi',132000);
```

## Output:

dev_id	country	experience	years	primary_language	secondary_language	database	used_framework	annual_salary *
1	5 Pakistan	1	c++	java	sqlite3	FastApi	None	3000
2	1 Pakistan	2	python	javascript	MySQL	Django	React	8000
3	19 India	3	javascript	HTML	PostgreSQL	Angular	Spring	12000
4	8 Pakistan	3	java	Kotlin	MySQL	Spring	Laravel	14000
5	17 Pakistan	5	C++	Python	MySQL	React	15000	15000
6	10 India	6	PHP	javascript	MySQL	Laravel	Spring	18000
7	2 India	5	java	SQL	Oracle	React	None	22000
8	14 Brazil	4	Python	javascript	MySQL	React	None	25000
9	30 UnitedArabOf	7	python	javascript	PostgreSQL	FastAPI	React	30000
10	7 Canada	4	javascript	Python	MongoDB	Vue	None	60000
11	11 France	8	Python	SQL	PostgreSQL	Django	None	65000
12	15 Netherlands	6	Go	Python	PostgreSQL	React	None	68000
13	4 Germany	7	C#	SQL	SQL Server	.NET	70000	70000
14	16 USA	3	Ruby	javascript	PostgreSQL	Rails	React	72000
15	12 Australia	9	javascript	Python	MongoDB	React	75000	75000
16	13 Japan	11	Java	Python	Oracle	Spring	None	80000
17	6 UK	12	Python	R	PostgreSQL	React	85000	85000
18	3 USA	10	javascript	TypeScript	PostgreSQL	React	90000	90000
19	18 Sweden	10	Python	julia	SQLite	Django	None	90000
20	9 USA	15	C++	Python	SQLite	Qt	None	110000
21	22 Vietnam	10	Java	PHP	MySQL	React	None	117000
22	24 India	23	C++	java	sqlite3	FastApi	None	131000
23	25 Bangladesh	7	python	sql	MySQL	React	132000	132000
24	21 Pakistan	15	C++	java	sqlite3	FastApi	None	133000
25	29 Pakistan	19	C++	java	sqlite3	FastApi	None	133000

## UPDATE PART:

### 1. Update salary of developers from Pakistan by +10%:

Command Input:

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is 'tech\_survey.db'. The menu bar includes File, Edit, View, Tools, and Help. The toolbar has buttons for New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL, with Execute SQL selected. A toolbar with various icons is located below the tabs. The main area contains a SQL editor window titled 'SQL 1' with the following code:

```
1 UPDATE dev_survey
2 SET annual_salary=annual_salary*1.1
3 WHERE country='Pakistan';
```

Below the SQL editor, the output window displays the results of the execution:

```
Execution finished without errors.
Result: query executed successfully. Took 0ms, 6 rows affected
At line 1:
UPDATE dev_survey
SET annual_salary=annual_salary*1.1
WHERE country='Pakistan';
```

### Command:

```
UPDATE dev_survey
SET annual_salary=annual_salary*1.1
WHERE country='Pakistan';
```

### Output:

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/

Table: dev\_survey

	dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
	Filter	Pak	Filter	Filter	Filter	Filter	Filter	Filter
1	5	Pakistan	1	C	NULL	NULL	NULL	3300.0
2	1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	8	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
4	17	Pakistan	5	C++	Python	MySQL	NULL	15400.0
5	21	Pakistan	19	c++	java	sqlite3	FastApi	146300
6	25	Pakistan	19	c++	java	sqlite3	FastApi	146300

Figure 3 Before

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/

Table: dev\_survey

	dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
	Filter	Pak	Filter	Filter	Filter	Filter	Filter	Filter
1	5	Pakistan	1	C	NULL	NULL	NULL	3630.0
2	1	Pakistan	2	Python	JavaScript	MySQL	Django	9680
3	8	Pakistan	3	Java	Kotlin	MySQL	Spring	14520.0
4	17	Pakistan	5	C++	Python	MySQL	NULL	16940.0
5	21	Pakistan	19	c++	java	sqlite3	FastApi	160930
6	25	Pakistan	19	c++	java	sqlite3	FastApi	160930

Figure 4 After

## 2. Update framework to NULL where database\_used is NULL:

**Command Input:**

The screenshot shows the DB Browser for SQLite interface. In the top menu, 'File', 'Edit', 'View', 'Tools', and 'Help' are visible. Below the menu, there are buttons for 'New Database', 'Open Database', 'Write Changes', 'Revert Changes', 'Open Project', 'Save Project', 'Attach Database', and 'Close Database'. The main window has tabs for 'Database Structure', 'Browse Data', 'Edit Pragmas', and 'Execute SQL'. The 'Execute SQL' tab is selected, showing a SQL editor with the following code:

```

1 UPDATE dev_survey
2 SET framework=NULL
3 WHERE database_used IS NULL;

```

Below the editor, a message box displays the results of the execution:

Execution finished without errors;  
1 rows were affected successfully. Took 0ms, 2 rows affected  
At line 1  
UPDATE dev\_survey  
SET framework=NULL  
WHERE database\_used IS NULL;

## Command:

UPDATE dev\_survey

SET framework=NULL

WHERE database\_used IS NULL;

## Output:

The screenshot shows the 'Browse Data' view of the 'dev\_survey' table in DB Browser for SQLite. The table has the following columns: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data is presented in a grid format with 25 rows. The first few rows are as follows:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	1	C	NULL	NULL	FLASK	3300.0
2	Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	India	2	JavaScript	HTML	Firebase	Angular	10000
4	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0

... (The table continues with 20 more rows of data.)

Figure 5 Before

The screenshot shows the DB Browser for SQLite interface with the 'tech\_survey.db' database open. The 'dev\_survey' table is selected, displaying 25 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various countries like Pakistan, India, Brazil, South Africa, Canada, France, Netherlands, Germany, USA, Australia, Japan, UK, Sweden, Vietnam, INDia, Bangladesh, and Pakistan. Languages listed include C, Python, JavaScript, Java, PHP, C++, SQL, Oracle, MySQL, PostgreSQL, MongoDB, Django, Angular, Spring, Flask, React, and TypeScript. Salaries range from 3300.0 to 146300.

	dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	5	Pakistan	1	C	NULL	NULL	NULL	3300.0
2	1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	19	India	2	JavaScript	HTML	Firebase	Angular	10000
4	8	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
5	10	India	6	PHP	JavaScript	MySQL	Laravel	15000
6	17	Pakistan	5	C++	Python	MySQL	NULL	15400.0
7	2	India	5	Java	SQL	Oracle	Spring	18000
8	14	Brazil	4	Python	JavaScript	MySQL	Flask	22000
9	20	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
10	7	Canada	4	JavaScript	Python	MongoDB	Vue	60000
11	11	France	8	Python	SQL	PostgreSQL	Django	65000
12	15	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
13	4	Germany	7	C#	SQL	SQL Server	.NET	70000
14	16	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000
15	12	Australia	9	JavaScript	NULL	MongoDB	React	75000
16	13	Japan	11	Java	Python	Oracle	Spring	82000
17	6	UK	12	Python	R	PostgreSQL	Flask	85000
18	3	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
19	18	Sweden	10	Python	Julia	SQLite	Django	90000
20	9	USA	15	C++	Python	SQLite	Qt	110000
21	22	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
22	24	INDia	23	C++	java	sqlite3	FastApi	131000
23	26	Bangladesh	7	c++	java	sqlite3	FastApi	132000
24	21	Pakistan	19	c++	java	sqlite3	FastApi	146300
25	25	Pakistan	19	c++	java	sqlite3	FastApi	146300

Figure 6 After

### 3. Change the primary language of dev\_id = 5 to Python:

**Command Input:**

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is 'tech\_survey.db'. The menu bar includes File, Edit, View, Tools, and Help. The toolbar contains icons for New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL, with Execute SQL selected. A toolbar below the tabs includes icons for file operations like Open, Save, Print, and Database. A central SQL editor window titled 'SQL 1' contains the following code:

```
1 UPDATE dev_survey
2 SET primary_language='python'
3 WHERE dev_id=5;
```

Below the SQL editor, the output pane displays the results of the executed query:

```
Execution finished without errors.
Result: query executed successfully. Took 0ms, 1 rows affected
At line 1:
UPDATE dev_survey
SET primary_language='python'
WHERE dev_id=5;
```

### Command:

```
UPDATE dev_survey
SET primary_language='python'
WHERE dev_id=5;
```

### Output:

DB Browser for SQLite - /home/aazan-noor-khu

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project

Database Structure Browse Data Edit Pragmas Execute SQL

Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary ▼ <sup>1</sup>
5	Pakistan	1	RUST	NULL	NULL	NULL	3300.0
1	15	Netherlands	6	Go	Python	PostgreSQL	68000
2	25	Pakistan	19	c++	java	sqlite3	FastApi
3							146300

Figure 7 Before

DB Browser for SQLite - /home/aazan-noor-khu

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project

Database Structure Browse Data Edit Pragmas Execute SQL

Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary ▼ <sup>1</sup>
5	Pakistan	1	python	NULL	NULL	NULL	3300.0
1	15	Netherlands	6	Go	Python	PostgreSQL	68000
2	25	Pakistan	19	c++	java	sqlite3	FastApi
3							146300

Figure 8 After

## DELETE PART:

1. Delete developers with experience < 2 years:

**Command Input:**

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is located at /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db. The menu bar includes File, Edit, View, Tools, and Help. The toolbar contains icons for New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL, with Execute SQL selected. A toolbar below the tabs includes icons for file operations like Open, Save, Print, and Database. A SQL editor window titled "SQL 1" contains the following code:

```
1 DELETE FROM dev_survey
2 WHERE (experience_years<2);
3
```

Below the SQL editor, the output pane displays the results of the executed query:

```
Execution finished without errors.
Result: query executed successfully. Took 0ms, 1 rows affected
At line 1:
DELETE FROM dev_survey
WHERE (experience_years<2);
```

### Command:

```
DELETE FROM dev_survey
```

```
WHERE (experience_years<2);
```

### Output:

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	1	python	NULL	NULL	NULL	3300.0
2	Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	India	2	JavaScript	HTML	Firebase	Angular	10000
4	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
5	India	6	PHP	JavaScript	MySQL	Laravel	15000
6	Pakistan	5	C++	Python	MySQL	NULL	15400.0
7	India	5	Java	SQL	Oracle	Spring	18000
8	Brazil	4	Python	JavaScript	MySQL	Flask	22000
9	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
10	Canada	4	JavaScript	Python	MongoDB	Vue	60000
11	France	8	Python	SQL	PostgreSQL	Django	65000
12	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
13	Germany	7	C#	SQL	SQL Server	.NET	70000
14	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000
15	Australia	9	JavaScript	NULL	MongoDB	React	75000
16	Japan	11	Java	Python	Oracle	Spring	82000
17	UK	12	Python	R	PostgreSQL	Flask	85000
18	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
19	Sweden	10	Python	Julia	SQLite	Django	90000
20	USA	15	C++	Python	SQLite	Qt	110000
21	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
22	INDia	23	c++	java	sqlite3	FastApi	131000
23	Bangladesh	7	c++	java	sqlite3	FastApi	132000
24	Pakistan	19	c++	java	sqlite3	FastApi	146300
25	Pakistan	19	c++	java	sqlite3	FastApi	146300

Figure 9 Before

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	India	2	JavaScript	HTML	Firebase	Angular	10000
3	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
4	India	6	PHP	JavaScript	MySQL	Laravel	15000
5	Pakistan	5	C++	Python	MySQL	NULL	15400.0
6	India	5	Java	SQL	Oracle	Spring	18000
7	Brazil	4	Python	JavaScript	MySQL	Flask	22000
8	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
9	Canada	4	JavaScript	Python	MongoDB	Vue	60000
10	France	8	Python	SQL	PostgreSQL	Django	65000
11	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
12	Germany	7	C#	SQL	SQL Server	.NET	70000
13	USA	3	Ruby	JavaScript	PostgreSQL	Rails	72000
14	Australia	9	JavaScript	NULL	MongoDB	React	75000
15	Japan	11	Java	Python	Oracle	Spring	82000
16	UK	12	Python	R	PostgreSQL	Flask	85000
17	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
18	Sweden	10	Python	Julia	SQLite	Django	90000
19	USA	15	C++	Python	SQLite	Qt	110000
20	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
21	INDia	23	c++	java	sqlite3	FastApi	131000
22	Bangladesh	7	c++	java	sqlite3	FastApi	132000
23	Pakistan	19	c++	java	sqlite3	FastApi	146300
24	Pakistan	19	c++	java	sqlite3	FastApi	146300

Figure 10 After

## 2. Delete developers whose salary is NULL:

## Command Input:

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is 'tech\_survey.db'. The toolbar includes File, Edit, View, Tools, Help, New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL, with Execute SQL selected. A toolbar below the tabs includes icons for file operations like Open, Save, and Print, along with a magnifying glass for search. A large central area is labeled 'SQL 1' with a close button. Inside this area, the following SQL code is written:

```
1 DELETE FROM dev_survey
2 WHERE annual_salary is NULL;
```

Below the SQL input area, the output window displays the results of the executed query:

```
Execution finished without errors.
Result: query executed successfully. Took 0ms, 3 rows affected
At line 1:
DELETE FROM dev_survey
WHERE annual_salary is NULL
```

## Command:

```
DELETE FROM dev_survey
```

```
WHERE annual_salary is NULL;
```

## Output:

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

File Edit View Tools Help  
 New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL  
 Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	India	19	JavaScript	HTML	Firebase	Angular	10000
3	Pakistan	8	Java	Kotlin	MySQL	Spring	13200.0
4	India	10	PHP	JavaScript	MySQL	Laravel	15000
5	Pakistan	17	C++	Python	MySQL	NULL	15400.0
6	India	2	Java	SQL	Oracle	Spring	NULL
7	Brazil	14	Python	JavaScript	MySQL	Flask	NULL
8	South Africa	20	Python	JavaScript	PostgreSQL	FastAPI	55000
9	Canada	7	JavaScript	Python	MongoDB	Vue	60000
10	France	11	Python	SQL	PostgreSQL	Django	65000
11	Netherlands	15	Go	Python	PostgreSQL	NULL	68000
12	Germany	4	C#	SQL	SQL Server	.NET	70000
13	USA	16	Ruby	JavaScript	PostgreSQL	Rails	NULL
14	Australia	12	JavaScript	NULL	MongoDB	React	75000
15	Japan	13	Java	Python	Oracle	Spring	82000
16	UK	6	Python	R	PostgreSQL	Flask	85000
17	USA	3	JavaScript	TypeScript	PostgreSQL	React	90000
18	Sweden	18	Python	Julia	SQLite	Django	90000
19	USA	9	C++	Python	SQLite	Qt	110000
20	Vietnam	22	NULL	PHP	NULL	NULL	117000
21	INDIA	24	c++	java	sqlite3	FastApi	131000
22	Bangladesh	26	c++	java	sqlite3	FastApi	132000
23	Pakistan	21	c++	java	sqlite3	FastApi	146300
24	Pakistan	25	c++	java	sqlite3	FastApi	146300

1 - 24 of 24 Go to: 1

Figure 11 Before

DB Browser for SQLite - /home/aazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db

File Edit View Tools Help  
 New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL  
 Table: dev\_survey Filter in any column

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	India	19	JavaScript	HTML	Firebase	Angular	10000
3	Pakistan	8	Java	Kotlin	MySQL	Spring	13200.0
4	India	10	PHP	JavaScript	MySQL	Laravel	15000
5	Pakistan	17	C++	Python	MySQL	NULL	15400.0
6	South Africa	20	Python	JavaScript	PostgreSQL	FastAPI	55000
7	Canada	7	JavaScript	Python	MongoDB	Vue	60000
8	France	11	Python	SQL	PostgreSQL	Django	65000
9	Netherlands	15	Go	Python	PostgreSQL	NULL	68000
10	Germany	4	C#	SQL	SQL Server	.NET	70000
11	Australia	12	JavaScript	NULL	MongoDB	React	75000
12	Japan	13	Java	Python	Oracle	Spring	82000
13	UK	6	Python	R	PostgreSQL	Flask	85000
14	USA	3	JavaScript	TypeScript	PostgreSQL	React	90000
15	Sweden	18	Python	Julia	SQLite	Django	90000
16	USA	9	C++	Python	SQLite	Qt	110000
17	Vietnam	22	NULL	PHP	NULL	NULL	117000
18	INDIA	24	c++	java	sqlite3	FastApi	131000
19	Bangladesh	26	c++	java	sqlite3	FastApi	132000
20	Pakistan	21	c++	java	sqlite3	FastApi	146300
21	Pakistan	25	c++	java	sqlite3	FastApi	146300

1 - 21 of 21 Go to: 1

Figure 12 After

## SELECT PART:

1. Display all developer records:

## Command:

```
SELECT * FROM dev_survey
```

## Output:

The screenshot shows the DB Browser for SQLite interface. The left pane displays the SQL query: `SELECT * FROM dev_survey`. The right pane shows the resulting table with 21 rows of data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various countries like Pakistan, USA, Germany, UK, Canada, etc., with their respective programming languages and salaries.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
3	Germany	7	C#	SQL	SQL Server	.NET	70000
4	UK	12	Python	R	PostgreSQL	Flask	85000
5	Canada	4	JavaScript	Python	MongoDB	Vue	60000
6	Pakistan	3	Java	Kotlin	MySQL	Spring	152000
7	USA	15	C++	Python	SQLite	Qt	110000
8	India	6	PHP	JavaScript	MySQL	Laravel	15000
9	France	8	Python	SQL	PostgreSQL	Django	65000
10	Australia	9	JavaScript	NULL	MongoDB	React	75000
11	Japan	11	Java	Python	Oracle	Spring	82000
12	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
13	Pakistan	5	C++	Python	MySQL	NULL	154000
14	Sweden	10	Python	Julia	SQLite	Django	90000
15	India	2	JavaScript	HTML	Firebase	Angular	10000
16	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
17	Pakistan	19	C++	java	sqlite3	FastAPI	146300
18	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
19	INDia	23	C++	java	sqlite3	FastAPI	131000
20	Pakistan	19	C++	java	sqlite3	FastAPI	146300

## 2. Display only country, primary\_language, annual\_salary:

### Command:

```
SELECT country,primary_language,annual_salary FROM dev_survey;
```

### Output:

The screenshot shows the DB Browser for SQLite interface. The left pane displays the SQL query: `SELECT country,primary_language,annual_salary FROM dev_survey`. The right pane shows the resulting table with 21 rows of data. The columns are: country, primary\_language, and annual\_salary. The data includes various countries like Pakistan, USA, Germany, UK, Canada, etc., with their respective primary languages and salaries.

country	primary_language	annual_salary
Pakistan	Python	8800
USA	JavaScript	90000
Germany	C#	70000
UK	Python	85000
Canada	JavaScript	60000
Pakistan	Java	132000
USA	C++	110000
India	PHP	15000
France	Python	65000
Australia	JavaScript	75000
Japan	Java	82000
Netherlands	Go	68000
Pakistan	C++	154000
Sweden	Python	90000
India	JavaScript	10000
South Africa	Python	55000
Pakistan	C++	146300
Vietnam	NULL	117000
INDia	C++	131000
Pakistan	C++	146300

### 3. List developers earning more than 50,000 USD:

#### Command:

```
SELECT * FROM dev_survey
```

```
WHERE (annual_salary>50000);
```

#### Output:

The screenshot shows the DB Browser for SQLite interface. The title bar reads "DB Browser for SQLite - /home/aaazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db". The main window has a toolbar with File, Edit, View, Tools, Help, New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL. The Execute SQL tab is active, showing a SQL editor with the following content:

```
1 SELECT * FROM dev_survey
2 WHERE (annual_salary>50000)
```

Below the SQL editor is a table with 16 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various countries like USA, Germany, UK, Canada, France, Australia, Japan, Netherlands, Sweden, South Africa, Pakistan, Vietnam, India, and Bangladesh, along with their respective developer details and salaries.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
2	Germany	7	C#	SQL	SQL Server	.NET	70000
3	UK	12	Python	R	PostgreSQL	Flask	85000
4	Canada	4	JavaScript	Python	MongoDB	Vue	60000
5	USA	15	C++	Python	SQLite	Qt	110000
6	France	8	Python	SQL	PostgreSQL	Django	65000
7	Australia	9	JavaScript	NULL	MongoDB	React	75000
8	Japan	11	Java	Python	Oracle	Spring	82000
9	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
10	Sweden	10	Python	Julia	SQLite	Django	90000
11	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
12	Pakistan	19	C++	java	sqlite3	FastAPI	146300
13	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
14	India	23	C++	java	sqlite3	FastAPI	131000
15	Pakistan	19	C++	java	sqlite3	FastAPI	146300
16	Bangladesh	7	C++	java	sqlite3	FastAPI	132000

At the bottom of the SQL editor, the message "Execution finished without errors. Result: 16 rows returned in 5ms At line 1: SELECT \* FROM dev\_survey WHERE (annual\_salary>50000)" is displayed. To the right of the SQL editor is a "Edit Database Cell" panel showing a single cell with the value "NULL". The status bar at the bottom right shows "UTF-8".

### 4. List developers with experience between 3 and 8 years:

#### Command:

```
SELECT * FROM dev_survey
```

```
WHERE experience_years BETWEEN 3 AND 8;
```

#### Output:

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the following code:

```
1 SELECT * FROM dev_survey
2 WHERE experience_years BETWEEN 3 AND 8;
```

The results table displays the following data:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Germany	7	C#	SQL	SQL Server	.NET	70000
2	Canada	4	JavaScript	Python	MongoDB	Vue	60000
3	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
4	India	6	PHP	JavaScript	MySQL	Laravel	15000
5	France	8	Python	SQL	PostgreSQL	Django	65000
6	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
7	Pakistan	5	C++	Python	MySQL	NULL	15400.0
8	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
9	Bangladesh	7	C++	java	sqlite3	FastAPI	132000

Execution finished without errors. Result: 9 rows returned in 6ms At line 1:  
SELECT \* FROM dev\_survey  
WHERE experience\_years BETWEEN 3 AND 8;

## 5. Show developers from Pakistan earning less than 20,000 USD:

**Command:**

```
SELECT * FROM dev_survey
```

```
WHERE country='Pakistan' AND annual_salary<20000;
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the following code:

```
1 SELECT * FROM dev_survey
2 WHERE country='Pakistan' AND annual_salary<20000;
```

The results table displays the following data:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
3	Pakistan	5	C++	Python	MySQL	NULL	15400.0

Execution finished without errors. Result: 3 rows returned in 6ms At line 1:  
SELECT \* FROM dev\_survey  
WHERE country='Pakistan' AND annual\_salary<20000;

## 6. Show developers who use Python and PostgreSQL:

**Command:**

```
SELECT *  
FROM dev_survey  
WHERE primary_language = 'Python' AND database_used = 'PostgreSQL';
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is located at /home/aaazan-noor-khuwaja/Aazan\_Data/4th\_Semester/DB\_Lab/lab2/tech\_survey.db. The main window has a toolbar with File, Edit, View, Tools, Help, New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database. Below the toolbar are tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL. The Execute SQL tab is active, containing the following SQL code:

```
1 SELECT *  
2 FROM dev_survey  
3 WHERE primary_language='Python' AND database_used='PostgreSQL';
```

Below the code is a table with the following data:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	UK	12	Python	R	PostgreSQL	Flask	85000
2	France	8	Python	SQL	PostgreSQL	Django	65000
3	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000

On the right side of the interface, there is a panel titled "Edit Database Cell" with the value "NULL". It includes fields for Mode (Text), Type of data currently in cell (NULL), and Identity (Select an identity to connect). Below this is a file browser showing a single file named "DBHub.io". At the bottom, there are tabs for SQL Log, Plot, DB Schema, and Remote, with the SQL Log tab selected.

**7. Show developers from USA or Germany earning above 60,000 USD:****Command:**

```
SELECT *  
FROM dev_survey  
WHERE country = 'USA' OR country = 'Germany' AND annual_salary>60000;
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. In the SQL tab, the following query is run:

```

1 SELECT *
2 FROM dev_survey
3 WHERE country = 'USA' OR country = 'Germany' AND annual_salary>60000;

```

The results table shows the following data:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
2	Germany	7	C#	SQL	SQL Server	.NET	70000
3	USA	15	C++	Python	SQLite	Qt	110000

The status bar at the bottom indicates: Execution finished without errors. Result: 3 rows returned in 6ms At Line 1: SELECT \*.

## 8. Show developers who do not use JavaScript:

**Command:**

`SELECT *`

`FROM dev_survey`

`WHERE primary_language != 'JavaScript';`

**Output:**

The screenshot shows the DB Browser for SQLite interface. In the SQL tab, the following query is run:

```

1 SELECT *
2 FROM dev_survey
3 WHERE primary_language != 'JavaScript';

```

The results table shows the following data:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	Germany	7	C#	SQL	SQL Server	.NET	70000
3	UK	12	Python	R	PostgreSQL	Flask	85000
4	Pakistan	3	Java	Kotlin	MySQL	Spring	132000
5	USA	15	C++	Python	SQLite	Qt	110000
6	India	6	PHP	JavaScript	MySQL	Laravel	15000
7	France	8	Python	SQL	PostgreSQL	Django	65000
8	Japan	11	Java	Python	Oracle	Spring	82000
9	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
10	Pakistan	5	C++	Python	MySQL	NULL	154000
11	Sweden	10	Python	Julia	SQLite	Django	90000
12	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
13	Pakistan	19	c++	java	sqlite3	FastAPI	NULL
14	India	23	c++	java	sqlite3	FastAPI	131000
15	Pakistan	19	c++	java	sqlite3	FastAPI	146300
16	Bangladesh	7	c++	java	sqlite3	FastAPI	132000

The status bar at the bottom indicates: Execution finished without errors. Result: 16 rows returned in 5ms At Line 1: SELECT \*.

## 9. Show developers with experience between 5 and 10 years:

**Command:**

```
SELECT *  
FROM dev_survey  
WHERE experience_years BETWEEN 5 AND 10;
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the following query:

```
1 SELECT *  
2 FROM dev_survey  
3 WHERE experience_years BETWEEN 5 AND 10;
```

The results pane displays a table with 10 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data is as follows:

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	3 USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
2	4 Germany	7	C#	SQL	SQL Server	.NET	70000
3	10 India	6	PHP	JavaScript	MySQL	Laravel	15000
4	11 France	8	Python	SQL	PostgreSQL	Django	65000
5	12 Australia	9	JavaScript	NULL	MongoDB	React	75000
6	15 Netherlands	6	Go	Python	PostgreSQL	NULL	68000
7	17 Pakistan	5	C++	Python	MySQL	NULL	15400.0
8	18 Sweden	10	Python	Julia	SQLite	Django	90000
9	20 South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
10	26 Bangladesh	7	c++	java	sqlite3	FastAPI	132000

The status bar at the bottom left indicates "Execution finished without errors. Result: 10 rows returned in 5ms". The status bar at the bottom right indicates "UTF-8".

## 10. Show developers whose primary language is either one of Python, Java, C++,PHP:

**Command:**

```
SELECT *  
FROM dev_survey  
WHERE primary_language IN ('Python','Java','C++','PHP');
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. In the SQL tab, the following query is run:

```

1 SELECT *
2 FROM dev_survey
3 WHERE primary_language IN ('Python', 'Java', 'C++', 'PHP');

```

The results table displays 10 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various combinations of languages and frameworks like Python-Django, Java-Spring, C++-Qt, etc., with salaries ranging from 65000 to 154000.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	UK	12	Python	R	PostgreSQL	Flask	85000
3	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
4	USA	15	C++	Python	SQLite	Qt	110000
5	India	6	PHP	JavaScript	MySQL	Laravel	15000
6	France	8	Python	SQL	PostgreSQL	Django	65000
7	Japan	11	Java	Python	Oracle	Spring	82000
8	Pakistan	5	C++	Python	MySQL	NULL	15400.0
9	Sweden	10	Python	Julia	SQLite	Django	90000
10	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000

Execution finished without errors.  
Result: 10 rows returned in 6ms  
At Line 1:  
SELECT \*

## 11. Show developers whose framework starts with S:

Command:

`SELECT *  
FROM dev_survey`

`WHERE framework LIKE 'S%';`

Output:

The screenshot shows the DB Browser for SQLite interface. In the SQL tab, the following query is run:

```

1 SELECT *
2 FROM dev_survey
3 WHERE framework LIKE 'S%';

```

The results table displays 2 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes Java-Spring and Oracle-Spring frameworks.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
2	Japan	11	Java	Python	Oracle	Spring	82000

Execution finished without errors.  
Result: 2 rows returned in 4ms  
At Line 1:  
SELECT \*

## 12. Show developers whose database contains the word SQL:

**Command:**

```
SELECT *  
FROM dev_survey  
WHERE database_used LIKE '%SQL%';
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the query: `SELECT * FROM dev_survey WHERE database_used LIKE '%SQL%'`. The results pane displays a table with 16 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various programming languages like Python, JavaScript, C#, R, Java, Go, PHP, and C++, along with databases like MySQL, PostgreSQL, and SQLite, and frameworks like Django, React, .NET, Flask, Spring, Laravel, Django, FastAPI, and SQLite3.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
3	Germany	7	C#	SQL	SQL Server	.NET	70000
4	UK	12	Python	R	PostgreSQL	Flask	85000
5	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
6	USA	15	C++	Python	SQLite	Qt	110000
7	India	6	PHP	JavaScript	MySQL	Laravel	15000
8	France	8	Python	SQL	PostgreSQL	Django	65000
9	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
10	Pakistan	5	C++	Python	MySQL	NULL	15400.0
11	Sweden	10	Python	Julia	SQLite	Django	90000
12	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
13	Pakistan	19	c++	java	sqlite3	FastAPI	NULL
14	India	23	c++	java	sqlite3	FastAPI	131000
15	Pakistan	19	c++	java	sqlite3	FastAPI	146300
16	Bangladesh	7	c++	java	sqlite3	FastAPI	132000

## 13. Show the top 5 highest-paid developers:

**Command:**

```
SELECT *  
FROM dev_survey  
ORDER BY annual_salary DESC  
LIMIT 5;
```

**Output:**

```

SELECT *
FROM dev_survey
ORDER BY annual_salary DESC
LIMIT 5;

```

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	25 Pakistan	19	C++	java	sqlite3	FastApi	146300
2	26 Bangladesh	7	C++	java	sqlite3	FastApi	132000
3	24 India	23	C++	java	sqlite3	FastApi	131000
4	22 Vietnam	NULL	NULL	PHP	NULL	NULL	117000
5	9 USA	15	C++	Python	SQLite	Qt	110000

Execution finished without errors.  
Result: 5 rows returned in 4ms  
At Line 1:  
SELECT \*

## 14. Show the lowest-paid developers sorted by salary:

**Command:**

```
SELECT *
```

```
FROM dev_survey
```

```
ORDER BY annual_salary ASC;
```

**Output:**

```

SELECT *
FROM dev_survey
ORDER BY annual_salary ASC;

```

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	21 Pakistan	19	C++	java	sqlite3	FastApi	NULL
2	1 Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	19 India	2	JavaScript	HTML	Firebase	Angular	10000
4	8 Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
5	10 India	6	PHP	JavaScript	MySQL	Laravel	15000
6	17 Pakistan	5	C++	Python	MySQL	NULL	15400.0
7	20 South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
8	7 Canada	4	JavaScript	Python	MongoDB	Vue	60000
9	11 France	8	Python	SQL	PostgreSQL	Django	65000
10	13 Netherlands	6	Go	Python	PostgreSQL	NULL	68000
11	4 Germany	7	C#	SQL	SQL Server	.NET	70000
12	12 Australia	9	JavaScript	NULL	MongoDB	React	75000
13	13 Japan	11	Java	Python	Oracle	Spring	82000
14	6 UK	12	Python	R	PostgreSQL	Flask	85000
15	3 USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
16	18 Sweden	10	Python	Julia	SQLite	Django	90000
17	9 USA	15	C++	Python	SQLite	Qt	110000
18	22 Vietnam	NULL	NULL	PHP	NULL	NULL	117000
19	24 India	23	C++	java	sqlite3	FastApi	131000

Execution finished without errors.  
Result: 19 rows returned in 4ms  
At Line 1:  
SELECT \*

## 15. Show the only developer with the most experience:

**Command:**

```
SELECT *  
FROM dev_survey  
ORDER BY experience DESC  
LIMIT 1;
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the following query:

```
1 SELECT *  
2 FROM dev_survey  
3 ORDER BY experience DESC  
4 LIMIT 1;
```

The results pane displays a table with 21 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various programming languages like Python, Java, C++, JavaScript, etc., and databases like MySQL, PostgreSQL, MongoDB, etc. The 'experience\_years' column shows values ranging from 2 to 23.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	19	c++	java	sqlite3	FastApi	NULL
2	Pakistan	2	Python	JavaScript	MySQL	Django	8800
3	India	2	JavaScript	HTML	Firebase	Angular	10000
4	Pakistan	3	Java	Kotlin	MySQL	Spring	13200.0
5	India	6	PHP	JavaScript	MySQL	Laravel	15000
6	Pakistan	5	C++	Python	MySQL	NULL	15400.0
7	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
8	Canada	4	JavaScript	Python	MongoDB	Vue	60000
9	France	8	Python	SQL	PostgreSQL	Django	65000
10	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
11	Germany	7	C#	SQL	SQL Server	.NET	70000
12	Australia	9	JavaScript	NULL	MongoDB	React	75000
13	Japan	11	Java	Python	Oracle	Spring	82000
14	UK	12	Python	R	PostgreSQL	Flask	85000
15	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
16	Sweden	10	Python	Julia	SQLite	Django	90000
17	USA	15	C++	Python	SQLite	Qt	110000
18	Vietnam	NULL	NULL	PHP	NULL	NULL	117000
19	India	23	c++	java	sqlite3	FastAPI	131000

The status bar at the bottom left indicates: Execution finished without errors. Result: 21 rows returned in 4ms. At Line 1: SELECT \*.

## 16. Show developers who use both primary and secondary languages:

**Command:**

```
SELECT *  
FROM dev_survey  
WHERE primary_language IS NOT NULL AND secondary_language IS NOT NULL;
```

**Output:**

```

SELECT *
FROM dev_survey
WHERE primary_language IS NOT NULL AND secondary_language IS NOT NULL;

```

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Pakistan	2	Python	JavaScript	MySQL	Django	8800
2	USA	10	JavaScript	TypeScript	PostgreSQL	React	90000
3	Germany	7	C#	SQL	SQL Server	.NET	70000
4	UK	12	Python	R	PostgreSQL	Flask	85000
5	Canada	4	JavaScript	Python	MongoDB	Vue	60000
6	Pakistan	3	Java	Kotlin	MySQL	Spring	132000
7	USA	15	C++	Python	SQLite	Qt	110000
8	India	6	PHP	JavaScript	MySQL	Laravel	15000
9	France	8	Python	SQL	PostgreSQL	Django	65000
10	Japan	11	Java	Python	Oracle	Spring	82000
11	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
12	Pakistan	5	C++	Python	MySQL	NULL	154000
13	Sweden	10	Python	Julia	SQLite	Django	90000
14	India	2	JavaScript	HTML	Firebase	Angular	10000
15	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
16	Pakistan	19	C++	java	sqlite3	FastAPI	NULL
17	INDIA	23	C++	java	sqlite3	FastAPI	131000
18	Pakistan	19	C++	java	sqlite3	FastAPI	146300
19	Bangladesh	7	C++	java	sqlite3	FastAPI	132000

Execution finished without errors.  
Result: 19 rows returned in 4ms  
At Line 1:  
SELECT \*

## 17. Show developers where salary is not provided:

**Command:**

`SELECT *  
FROM dev_survey`

`WHERE annual_salary IS NULL;`

**Output:**

```

SELECT *
FROM dev_survey
WHERE annual_salary IS NULL;

```

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
21	Pakistan	19	C++	Java	sqlite3	FastAPI	NULL

Execution finished without errors.  
Result: 1 rows returned in 5ms  
At Line 1:  
SELECT \*

## 18. Rank developers by salary within each country (sort based on two columns):

**Command:**

```
SELECT *  
FROM dev_survey  
ORDER BY country ASC,annual_salary DESC;
```

**Output:**

The screenshot shows the DB Browser for SQLite interface. The SQL tab contains the following query:

```
1 SELECT *  
2 FROM dev_survey  
3 ORDER BY country ASC,annual_salary DESC;
```

The results pane displays a table with 19 rows of developer survey data. The columns are: dev\_id, country, experience\_years, primary\_language, secondary\_language, database\_used, framework, and annual\_salary. The data includes various countries like Australia, Bangladesh, Canada, France, Germany, India, Netherlands, Pakistan, Sweden, UK, and USA, along with their respective developer details and salaries.

dev_id	country	experience_years	primary_language	secondary_language	database_used	framework	annual_salary
1	Australia	9	JavaScript	NULL	MongoDB	React	75000
2	Bangladesh	7	c++	java	sqlite3	FastApi	132000
3	Canada	4	JavaScript	Python	MongoDB	Vue	60000
4	France	8	Python	SQL	PostgreSQL	Django	65000
5	Germany	7	C#	SQL	SQL Server	.NET	70000
6	India	24	c++	java	sqlite3	FastApi	131000
7	India	6	PHP	JavaScript	MySQL	Laravel	15000
8	India	2	JavaScript	HTML	Firebase	Angular	10000
9	Japan	11	Java	Python	Oracle	Spring	82000
10	Netherlands	6	Go	Python	PostgreSQL	NULL	68000
11	Pakistan	19	c++	java	sqlite3	FastApi	146300
12	Pakistan	5	C++	Python	MySQL	NULL	15400.0
13	Pakistan	8	Java	Kotlin	MySQL	Spring	13200.0
14	Pakistan	1	Python	JavaScript	MySQL	Django	8800
15	Pakistan	21	c++	java	sqlite3	FastApi	NULL
16	South Africa	7	Python	JavaScript	PostgreSQL	FastAPI	55000
17	Sweden	10	Python	Julia	SQLite	Django	90000
18	UK	6	Python	R	PostgreSQL	Flask	85000
19	USA	9	C++	Python	SQLite	Qt	110000

Execution finished without errors.  
Result: 21 rows returned in 4ms  
All in line 1:  
SELECT \*