12 ஆம் வகுப்பு – கணினி அறிவியல்

பாடம் 15.

SQL மூலம் தரவுகளைக் கையாளுதல்

Prepared by,

J. KAVITHA, B.Sc,B.Ed,M.C.A,M.Phil.,

Computer Instructor Gr - I,

GHSS, S.S.KULAM,

COIMBATORE – 641107.

கற்றலின் நோக்கங்கள்

- ஒரு தரவு அட்டவணையை உருவாக்கி அதில் புதிய பதிவுகளைத் தரவுத்தளத்தில் சேர்த்தல்.
- ஒரு தரவு அட்டவணையில் உள்ள பதிவுகளை மேம்படுத்துதல் மற்றும் நீக்குதல்.
- ஒரு தரவு அட்டவணையில் வினவல்.
- CSV கோப்பில் வினவலை எழுதுதல்.

அறிமுகம்

- தரவுத்தளம் என்பது ஒருங்கிணைக்கப்பட்ட தரவுகளின் தொகுப்பாகும்.
- தரவுத்தளம் என்ற கூறு தரவுகளை அல்லது தரவுத்தள மேலாண்மை அமைப்பைக் குறிக்கும்.
- பயனர்கள் மற்றும் தரவுத்தளத்திற்கு இடையே உள்ள தொடர்புக்கு பயன்படும் மென்பொருள் பயன்பாடே தரவுத்தள மேலாண்மை அமைப்பாகும்.
- பயனர்கள் மனிதர்களாக இருக்க வேண்டிய அவசியமில்லை , பிற நிரல்களாகவோ அல்லது பயன்பாடுகளாகவோ இருக்கலாம்.

SQLITE

- SQLite என்பது எளிய உறவுநிலை தரவுத்தள அமைப்பாகும்.
- இது தரவுகளை முறையான தரவு கோப்புகளாக கணினியின் உட்புற நினைவகத்தில் சேமித்து வைக்கும்.

நன்மைகள்:

- Mysql அல்லது Oracle போல் இல்லாமல் உள்ளிணைந்த பயன்பாடாக வடிவமைக்கப்பட்டுள்ளது.
- வேகமாகவும், மிகுந்த சோதிக்கப்பட்டதாகவும் மற்றும் நெகிழ்வானதாகவும் உள்ளதால் SQLite-ல் வேலை செய்வது எளிதாகும்.

SQLite ஐ பயன்படுத்துதல்

- படிநிலை 1: sqlite3 ஐ இணைக்கவும்.
- படிநிலை 2: connect( ) வழிமுறையைப் பயன்படுத்தி இணைப்பை உருவாக்கி தரவுத்தளத்தின் பெயரை அணுகவும்.
- படிநிலை 3: cursor =connection.cursor( ) என்றக் கூற்றைப் பயன்படுத்தி cursor என்னும் பொருளை அணுகவும்.

Cursor:

- தரவுத்தளத்திலுள்ள cursor என்பது தரவுத்தள பதிவுகளின் மீது செயல்படும் ஒரு கட்டுப்பாட்டு அமைப்பாகும். SQL –ன் அனைத்து கட்டளைகளையும் செயல்படுத்த இது பயன்படுகிறது.

SQLite டைப் பயன்படுத்தி தரவுத்தளத்தை உருவாக்குதல்

நிரல்

```
import sqlite3
connection = sqlite3.connect ("Academy.db")
cursor = connection.cursor()
```

- "Academy" என்ற பெய ரில் தரவுத்தளம் உருவாக்கப்பட்டுள்ளது. இது SQL-ன் "CREATE DATABASE Academy;" என்ற கட்டளைக்கு இணையானது.
- "sqlite3. connect ('Academy.db')" என்ற கூற்றை அழைக்கும் போது ஏற்கனவே உருவாக்கிய தரவுத்தளத்தைத் திறக்கும்.

அட்டவணையை உருவாக்குதல்

- ஒரு வெற்று தரவுத்தளத்தை உருவாக்கிய பிறகு, முடிந்தவரை ஒன்று அல்லது அதற்கு மேற்பட்ட அட்டவணைகளைச் சேர்க்கலாம்.

கட்டளை அமைப்பு

```
CREATE TABLE Student (Rollno INTEGER PRIMARY KEY,
      Sname VARCHAR(20), Grade CHAR(1), gender CHAR(1),
      Average DECIMAL(5,2), birth_date DATE);
```

- அட்டவணையிலுள்ள தரவு ஒற்றை அல்லது இரட்டை மேற்கோள்குறியுடன் உள்ளதால், பைத்தானிலுள்ள SQL கட்டளைகள் மூன்று மேற்கோள் குறியினால் குறிக்கப்படும்.

புலத்தை “INTEGER PRIMARY KEY” என அறிவிப்பதன் நன்மை

- அட்டவணையில் உள்ள ஒரு நெடுவரிசை **INTEGER PRIMARY KEY,** என அறிவிக்கப்பட்டு, எப்பொழுதெல்லாம் NULL என்ற மதிப்பு உள்ளீடாக பயன்படுத்தப்படுகிறதோ, அந்த NULL மதிப்பு தானாகவே அந்த நெடுவரிசையில் இது வரை பயன்படுத்தப்பட்ட மிக உயர்ந்த மதிப்பைவிட ஒன்று மிகுந்து முழு எண்ணாக இருக்கும்.
- வெற்று அட்டவணை எனில் 1 என்ற மதிப்பு பயன்படுத்தப்படும்.

பதிவுகளைச் சேர்த்த ல் (Adding Records):

- INSERT கட்டளையை SQLite ல் அனுப்புவதன் மூலம் அட்டவணையில் தரவுகளை சேர்க்கலாம்.
- execute() செயற்கூறு கொடுக்கப்பட்ட SQL கட்டளையை செயல்டுத்தும்.

எடுத்துக்காட்டு நிரல்1:

```
import sqlite3
connection = sqlite3.connect ("Academy.db")
cursor = connection.cursor()
#cursor.execute ("""DROP TABLE Student;""")
sql_command = """
CREATE TABLE Student (
Rollno INTEGER PRIMARY KEY , Sname VARCHAR(20), Grade CHAR(1),
gender CHAR(1), Average DECIMAL (5, 2), birth_date DATE);"""
cursor.execute(sql_command)
      sql_command = """INSERT INTO Student (Rollno, Sname, Grade, gender,
      Average, birth_ date) VALUES (NULL, "Akshay", "B", "M","87.8", "2001-12-
      12");""" cursor.execute(sql_ command)
      sql_command = """INSERT INTO Student (Rollno, Sname, Grade, gender,
      Average, birth_date) VALUES (NULL, "Aravind", "A", "M","92.50","2000-08-
      17");""" cursor. execute(sql_command)
connection.commit()
connection.close()
print("STUDENT TABLE CREATED")
```

வெளியீடு

```
STUDENT TABLE CREATED
```

எடுத்துக்காட்டு நிரல்2:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
student_data = [("Baskar","C","M","75.2","1998-05-17"),
("Sajini","A","F","95.6","2002-11-01"),
("Varun","A","M","80.6","2002-02-01"),
("Priya","A","F","98.6","2002-03-14"),
("Tarun","D","M","62.3","1999-02-19")]
for p in student_data:
    format_str = """INSERT INTO Student (Rollno, Sname, Grade, gender,Average,
    birth_date) VALUES (NULL,"{name}", "{gr}", "{gender}","{avg}","{birthdate}");"""
     sql_command=(format_str.format(name= p[0], gr=p[1], gender=p[2], avg=p[3],
     birthdate = p[4]))
    cursor.execute(sql_command)
connection.commit()
connection.close()
print("RECORDS ADDED TO STUDENT TABLE ")
```

வெளியீடு

```
RECORDS ADDED TO STUDENT TABLE
```

பைத்தானை பயன்படுத்தி SQL வினவல்

- Select கூற்று SQL-ல் மிகவும் பொதுவாக பயன்படுத்தக்கூடிய கூற்று ஆகும். இந்த கூற்று தரவுத்தளத்திலுள்ள அட்டவணையிலிருந்து தரவுகளைப் பெற பயன்படுகிறது.
- **கட்டளை அமைப்பு: Select * from table_name**

எடுத்துக்காட்டு நிரல்:

```
#save the file as “sql_Academy_query.py”
import sqlite3
connection = sqlite3.connect("Academy.db")
# cursor object
crsr = connection.cursor()
 # execute the command to fetch all the data from the table Student
crsr.execute("SELECT * FROM Student")
# store all the fetched data in the ans variable
ans= crsr.fetchall()
# loop to print all the data
for i in ans:
print(i)
```

FETCHALL() செயற்கூறு

- fetchall() செயற்கூறு அனைத்து வரிசைகளையும் தரவுத்தள அட்டவணையில் இருந்து பெற பயன்படுகிறது.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT * FROM student")
print("fetchall:")
result = cursor.fetchall()
for r in result:
print(r)
```

வெளியீடு

```
fetchall:
(1, 'Akshay', 'B', 'M', 87.8, '2001-12-12')
(2, 'Aravind', 'A', 'M', 92.5, '2000-08-17')
(3, 'BASKAR', 'C', 'M', 75.2, '1998-05-17')
(4, 'SAJINI', 'A', 'F', 95.6, '2002-11-01')
(5, 'VARUN', 'B', 'M', 80.6, '2001-03-14')
(6, 'PRIYA', 'A', 'F', 98.6, '2002-01-01')
(7, 'TARUN', 'D', 'M', 62.3, '1999-02-01')
```

FETCHONE() செயற்கூறு

- fetchone() செயற்கூறு வினவல் முடிவுத் தொகுதியின் உள்ளே உள்ள அடுத்த வரிசையைக் கொடுக்கும் (அல்லது) எந்த வரிசையும் இல்லை என்றால் None என்ற மதிப்பை கொடுக்கும்.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT * FROM student")
print("\nfetch one:")
res = cursor.fetchone()
print(res)
```

வெளியீடு

```
fetch one:
(1, 'Akshay', 'B', 'M', 87.8, '2001-12-12')
```

FETCHMANY() செயற்கூறு

- குறிப்பிட்ட எண்ணிக்கையிலான பதிவுகளைக் காண்பிக்க fetchmany() செயற்கூறு பயன்படுகிறது. இந்த செயற்கூறு முடிவுத் தொகுதியில் மீதம் உள்ள வரிசைகளின் எண்ணிக்கையைக் கொடுக்கும்.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT * FROM student")
print("fetching first 3 records:")
result = cursor.fetchmany(3)
print(result)
```

வெளியீடு

```
fetching first 3 records:
[(1, 'Akshay', 'B', 'M', 87.8, '2001-12-12'),
(2, 'Aravind', 'A', 'M', 92.5, '2000-08-17'),
(3, 'BASKAR', 'C', 'M', 75.2, '1998-05-17')]
```

SQL-ல் துணைநிலை கூற்று (CLAUSES)

- SQL வழங்கும் பல வகையான துணைநிலை கூற்றுகள் SELECT கூற்றுகளில் பயன்படுத்தப்படுகிறது.
  - SQL DISTINCT துணைநிலை கூற்று
  - SQL WHERE துணைநிலை கூற்று
  - GROUP BY துணைநிலைக்கூற்று
  - ORDER BY துணைநிலைக்கூற்று
  - Having துணைநிலைக்கூற்று

SQL DISTINCT துணைநிலை கூற்று

- ஒரு குறிப்பிட்ட நெடுவரிசை அல்லது அட்டவணையில் உள்ள இரட்டிப்பு மதிப்புகளைத் தவிர்ப்பதற்காக DISTINCT துணைநிலை கூற்று பயன்படுகிறது. இந்த சிறப்பு சொல்லைப் பயன்படுத்தும் போது தனித்த மதிப்புகளை பெற முடியும்.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT DISTINCT (Grade) FROM student")
result = cursor.fetchall()
print(result)
```

வெளியீடு

```
[('B',), ('A',), ('C',), ('D',)]
```

SQL WHERE துணைநிலை கூற்று

- குறிப்பிட்ட நிபந்தனைகளை நிறைவேற்றும் பதிவுகளை மட்டுமே பிரித்தெடுக்க **WHERE துணை நிலை கூற்று** பயன்படுகிறது.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT DISTINCT (Grade) FROM student
where gender='M'")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
    ('B',)
    ('A',)
    ('C',)
    ('D',)
```

GROUP BY துணைநிலைக்கூற்று

- குறிப்பிட்ட பதிவுகளைச் சுருக்கமான வரிசைகளைக் கொண்ட குழுவாக சேர்க்கிறது. இது ஒவ்வொரு குழுவிற்கும் ஒரு பதிவை கொடுக்கிறது.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT gender,count(gender) FROM student Group BY gender")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
    ('F', 2)
    ('M', 5)
```

ORDER BY துணைநிலைக்கூற்று

- SELECT கூற்றுடன் சேர்ந்து குறிப்பிட்ட புலங்களில் உள்ள தரவுகளை முறையாக வரிசையாக்கம் செய்ய பயன்படுகிறது.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT Rollno,sname FROM student Order BY sname")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
(1, 'Akshay')
(2, 'Aravind')
(3, 'BASKAR')
(6, 'PRIYA')
(4, 'SAJINI')
(7, 'TARUN')
(5, 'VARUN')
```

HAVING துணைநிலைக்கூற்று

- GROUP BY செயற்கூறின் அடிப்படையில் தரவுகளை வடிகட்ட பயன்படுகிறது. இது Where கூற்றை ஒத்ததாகும். ஆனால் குழு சார்புகளுடன் பயன்படுகிறது. Where துணைநிலைக்கூற்று ‘Group by’ துணைநிலைக்கூற்றுடன் பயன்படுத்த முடியாது.

எடுத்துக்காட்டு:

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT GENDER,COUNT(GENDER) FROM Student GROUP
BY GENDER HAVING COUNT(GENDER)>3")
result = cursor.fetchall()
co = [i[0] for i in cursor.description]
print(co)
print(result)
```

வெளியீடு

```
['gender', 'COUNT(GENDER)']
[('M', 5)]
```

SQL AND, OR மற்றும் NOT செயற்குறிகள்

- WHERE துணைநிலைக்கூற்று AND, OR, மற்றும் NOT செயற்குறிகளுடன் இணைத்துப் பயன்படுத்தலாம்.
- ஒன்றிற்கு மேற்பட்ட நிபந்தனைகளின் அடிப்படையில் பதிவுகளை வடிகட்ட ‘AND’ மற்றும் ‘OR’ செயற்குறிகள் பயன்படுகின்றன.

AND செயற்குறியுடன் உள்ள எடுத்துக்காட்டு

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT Rollno,Sname,Average FROM student
WHERE (Average>=80 AND Average<=90)")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
    (1, 'Akshay', 87.8)
    (5, 'VARUN', 80.6)
```

SQL AND, OR மற்றும் NOT செயற்குறிகள்

OR செயற்குறியுடன் உள்ள எடுத்துக்காட்டு

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT Rollno,sname FROM student WHERE
(Average<60 OR Average>70)")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
    (1, 'Akshay')
    (2, 'Aravind')
    (3, 'BASKAR')
    (4, 'SAJINI')
    (5, 'VARUN')
   (6, 'PRIYA')
```

SQL AND, OR மற்றும் NOT செயற்குறிகள்

NOT செயற்குறியுடன் உள்ள எடுத்துக்காட்டு

```
import sqlite3
connection = sqlite3.connect("Academy.db")
cursor = connection.cursor()
cursor.execute("SELECT * FROM student where NOT grade ='A' and NOT Grade='B'")
result = cursor.fetchall()
print(*result,sep="\n")
```

வெளியீடு

```
    (3, 'BASKAR', 'C', 'M', 75.2, '1998-05-17')
    (7, 'TARUN', 'D', 'M', 62.3, '1999-02-01')
```

மதிப்பீட்டுச் சார்புகள்

- மதிப்பீட்டுச் சார்புகள் நெடுவரிசையில் உள்ள மதிப்புகளைக் கொண்டு செயல்பாடுகளைச் செய்து ஒரே ஒரு மதிப்பை விடையாகக் கொடுக்கும்.

COUNT() சார்பு	**அட்டவணையிலுள்ள வரிசைகளின் எண்ணிக்கையை திருப்பி அனுப்பும்.**
AVG() சார்பு	அட்டவணையிலுள்ள தேர்ந்தெடுக்கப்பட்ட புலத்தின் பதிவுகளின் சராசரியை கணக்கிட பயன்படுகிறது.
SUM() சார்பு	அட்டவணையிலுள்ள தேர்ந்தெடுக்கப்பட்ட புலத்தின் பதிவுகளின் கூட்டுத்தொகையைக் கணக்கிடுகிறது.
MAX() சார்பு	தேர்ந்தெடுக்கப்பட்ட புலத்தின் பெரிய மதிப்பை திருப்பி அனுப்பும்.
MIN() சார்பு	தேர்ந்தெடுக்கப்பட்ட புலத்தின் சிறிய மதிப்பை திருப்பி அனுப்பும்.

பிற செயல்பாடுகள்

- **பதிவுகளைப் புதுப்பித்தல்:** பைத்தான் ஸ்கிரிப்ட் மூலமாக பதிவுகளைப் புதுப்பிக்கமுடியும். இதற்கு UPDATE () செயற்கூறு பயன்படுகிறது.
- **நீக்குதல் செயல்பாடு:** SQL ல் உள்ள DELETE () கட்டளைகளைக் கொண்டு பதிவுகளை நீக்குவது போல பைத்தானிலும் நீக்க முடியும்.
- **பயனரால் உள்ளிடப்படும் தரவு:** இயங்கு நேரத்தில் பைத்தான் input() கட்டளையைப் பயன்படுத்தி தரவுகளை உள்ளீடு செய்ய முடியும்.

பிற செயல்பாடுகள்

- **வினவலை CSV கோப்புடன் ஒருங்கிணைத்தல்**: வினவலின் வெளியீட்டை“SQL.CSV” என்னும் CSV கோப்பில் எழுதலாம். இந்த கோப்பினை MS-Excel மூலம் திறந்து முடிவுகளை காணலாம் அல்லது ஸ்கிரிப்ட்டைப் பயன்படுத்தியும் CSV கோப்புகளைத் திறந்து முடிவுகளைக் காணலாம்.
- **sqlite_master:** sqlite_master என்பது முதன்மை அட்டவணையாகும். இது நமது தரவுத்தள அட்டவணைகளின் முக்கிய தகவல்களைக் கொண்டிருக்கும்.
- **பைத்தானில் கோப்பின் பாதை:** பைத்தானில், கோப்பின் பாதையை ‘/’ அல்லது ‘\\’ குறியீடு குறிக்கப்படுகிறது. எடுத்துக்காட்டாக, பாதையை 'c:/pyprg/sql.csv', அல்லது c:\\pyprg\\sql.csv’ எனக் குறிப்பிடலாம்.

முக்கிய வினாக்கள்

1. தரவுத்தளத்தை பயன்படுத்தும் பயனர்களைக் குறிப்பிடவும்.
2. தரவுத்தளத்தை இணைக்க பயன்படும் முறைகள் யாவை? எ.கா. தருக.
3. SQLite என்றால் என்ன? இதன் நன்மைகள் யாவை?
4. புலத்தை “INTEGER PRIMARY KEY” என அறிவிப்பதன் நன்மை என்ன?
5. அட்டவணையில் பதிவுகளை விரிவுப்படுத்துவதற்கான கட்டளையை எழுதுக. எ.கா தருக.
6. தரவுத்தள அட்டவணையிலிருந்து அனைத்து பதிவுகளையும் பெறுவதற்கான வழிமுறை எது?
7. fetch one( ) மற்றும் fetch many( ) வேறுபடுத்துக.
8. Where துணைநிலைக்கூற்றின் பயன் என்ன? where கூற்றைப் பயன்படுத்தி ஒரு பைத்தான் கூற்றை எழுதவும்.
9. HAVING துணை நிலைக்கூற்றின் பயன் யாது? எடுத்துக்காட்டு தருக.

நன்றி!

கல்வி என்பது கடல்.
அதை கற்றுக் கொடுப்பது
தொழில் அல்ல தவம்.
விருப்பம் பல கொண்டு
விரைவுடன் நீ படித்தால்
வாழ்வில் மேன்மை
பெறலாம்..

வாழ்த்துக்கள்.

ஜெ. கவிதா B.Sc, B.Ed, M.C.A, M.Phil.,
கணினி பயிற்றுநர் நிலை - I
அரசு மேல்நிலைப்பள்ளி,
சர்க்கார்சாமக்குளம்,
கோயம்புத்தூர் – 641107.