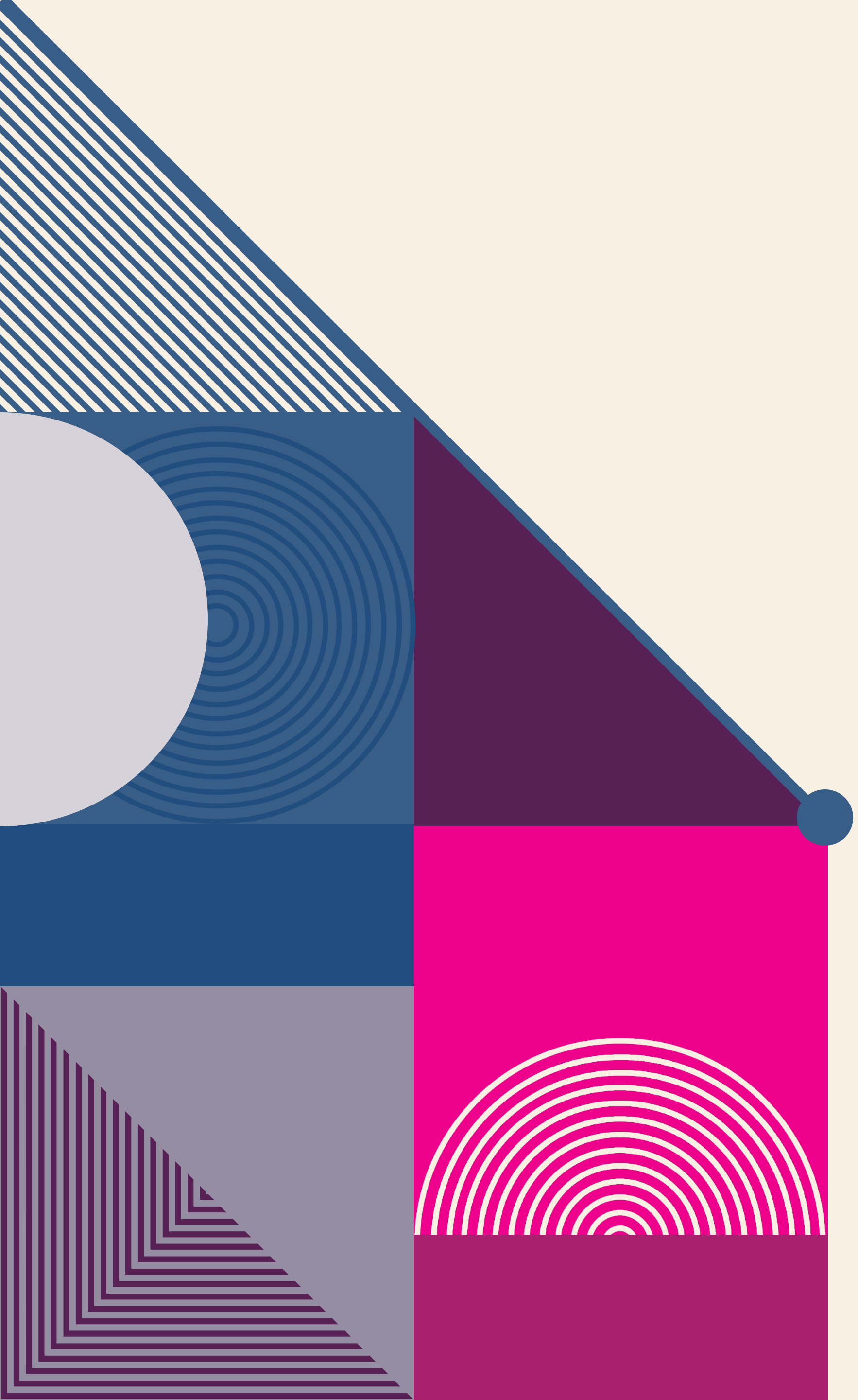
PRESENTATION M1&L1

TABLE OF CONTENTS

- Introduction and Relational Databases & SQL
- Test Driven Development
- How cloud computing helps business ?
- How analytics help businesses ?
- OWASP Top 10
- SQL Analysis Task
- 12 Factors
- Agile Manifesto

INTRODUCTION TO RDBMS & SQL



RELATIONAL DATABASE

1. A relational database is a type of database that organizes data into rows and columns, which collectively form a table where the data points are related to each other.
2. Relational databases are also typically associated with transactional databases, which execute commands, or transactions, collectively. A popular example that is used to illustrate this is a banking system

RELATIONAL DATABASE

And Transaction cannot occur in any kind of partial state and transaction properties is defined by the ACID

Atomicity

Consistency

Solation

Durability

SQL

SQL stands for Structured Query Language and it is used to interact with the RDBMS and allows efficient querying, updating, and management of relational databases.

IMPORTANT COMPONENTS OF SQL

1. Stored procedures
2. Transactions
3. Views
4. Indexes
5. Joins
6. CTE(Coman Table Expression)

SQL COMMANDS

Data Definition Language(DDL)

1. CREATE
2. ALTER
3. DROP

SQL COMMANDS

Data Manipulation Language(DML)

1. INSERT
2. UPDATE
3. DELETE

SQL COMMANDS

Data Control language (DCL)

1. **GRANT**
2. **REVOKE**

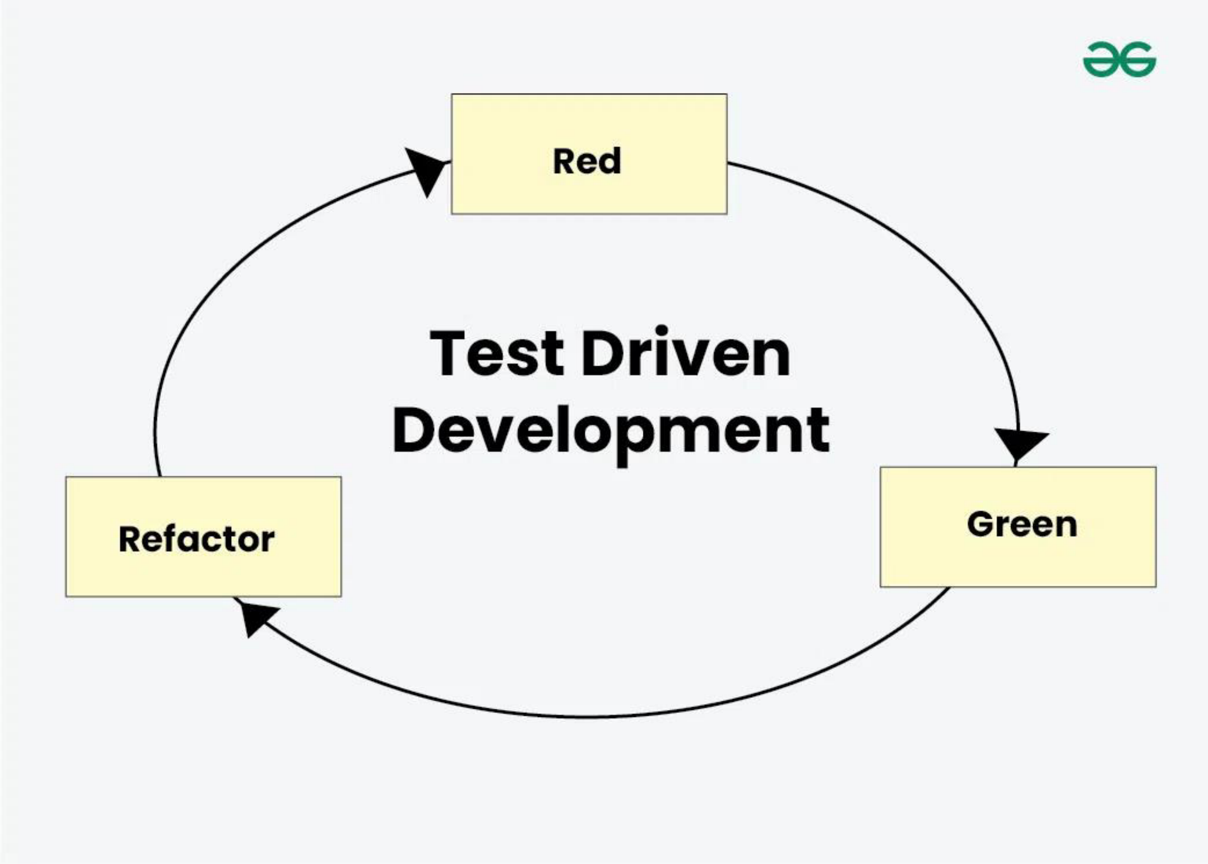
Transaction Control Language(TCL)

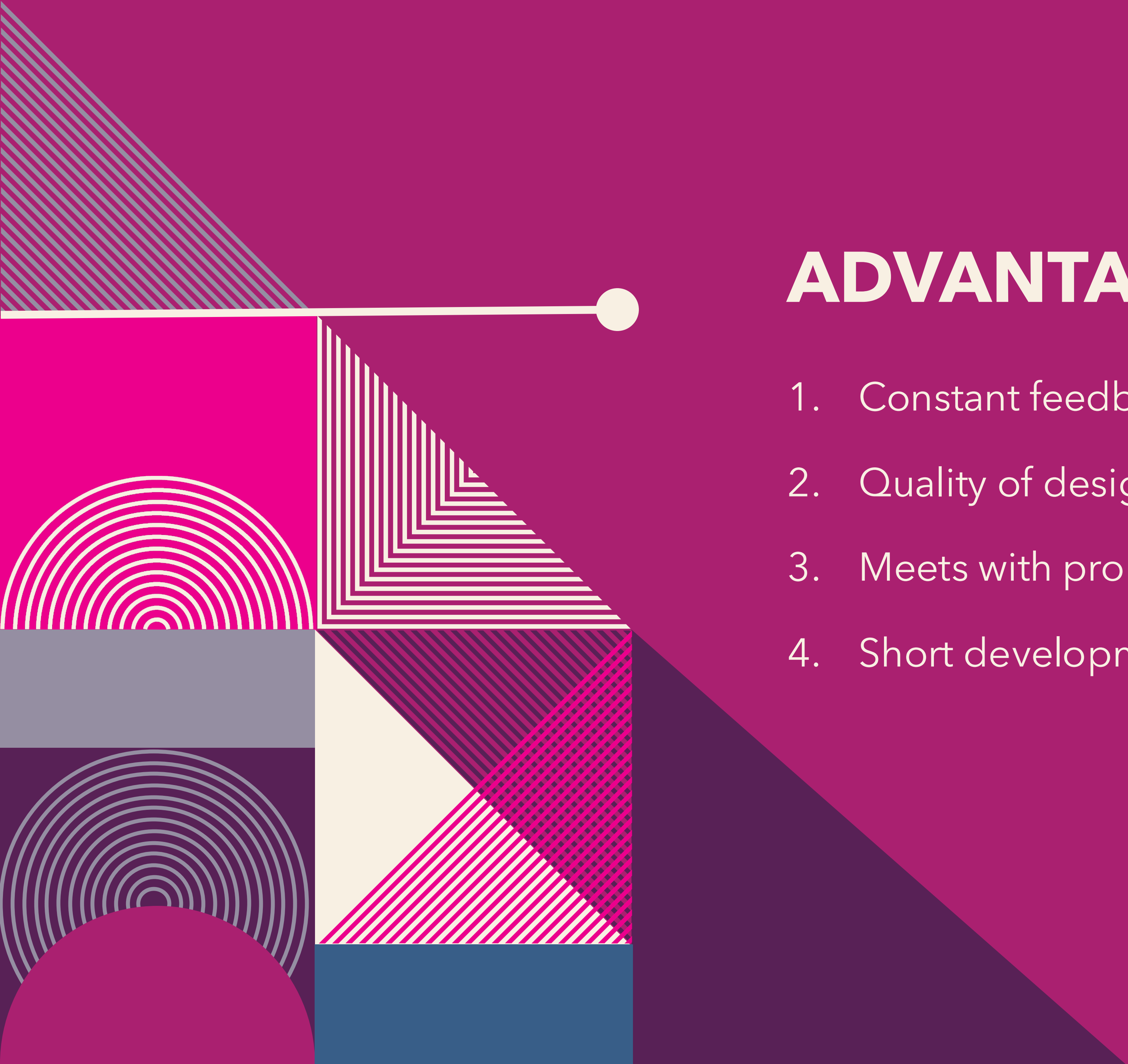
1. **COMMIT**
2. **ROLLBACK**

TEST DRIVEN DEVELOPMENT

Test Driven Development (TDD) is a software development methodology that emphasizes writing **tests before writing the actual code**.

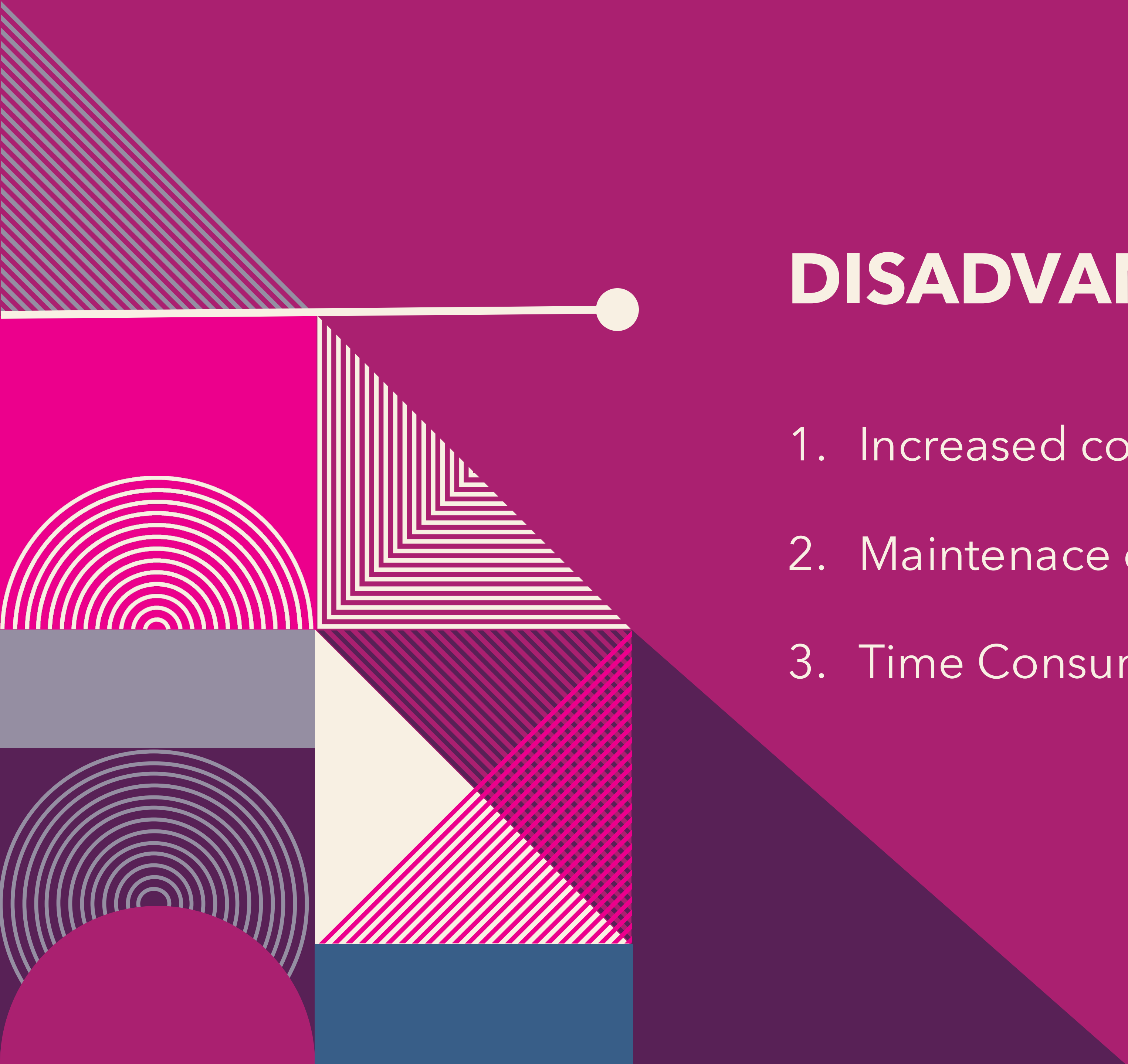
PROCESS OF TEST DRIVEN DEVELOPMENT (TDD)





ADVANTAGE OF TDD

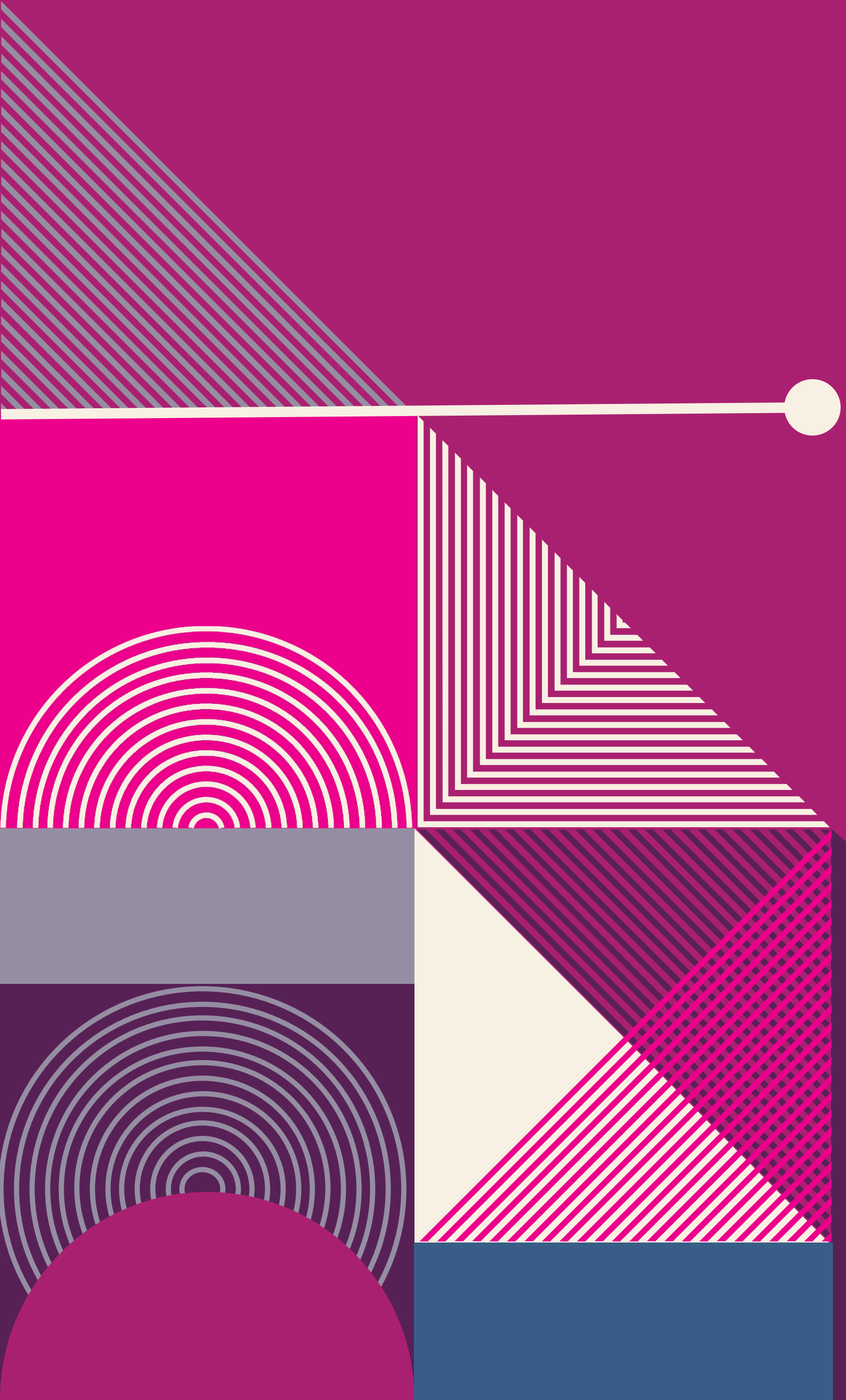
1. Constant feedback
2. Quality of design
3. Meets with proper requirement
4. Short development lifecycle



DISADVANTAGE OF TDD

1. Increased code volume
2. Maintenace overheads
3. Time Consuming

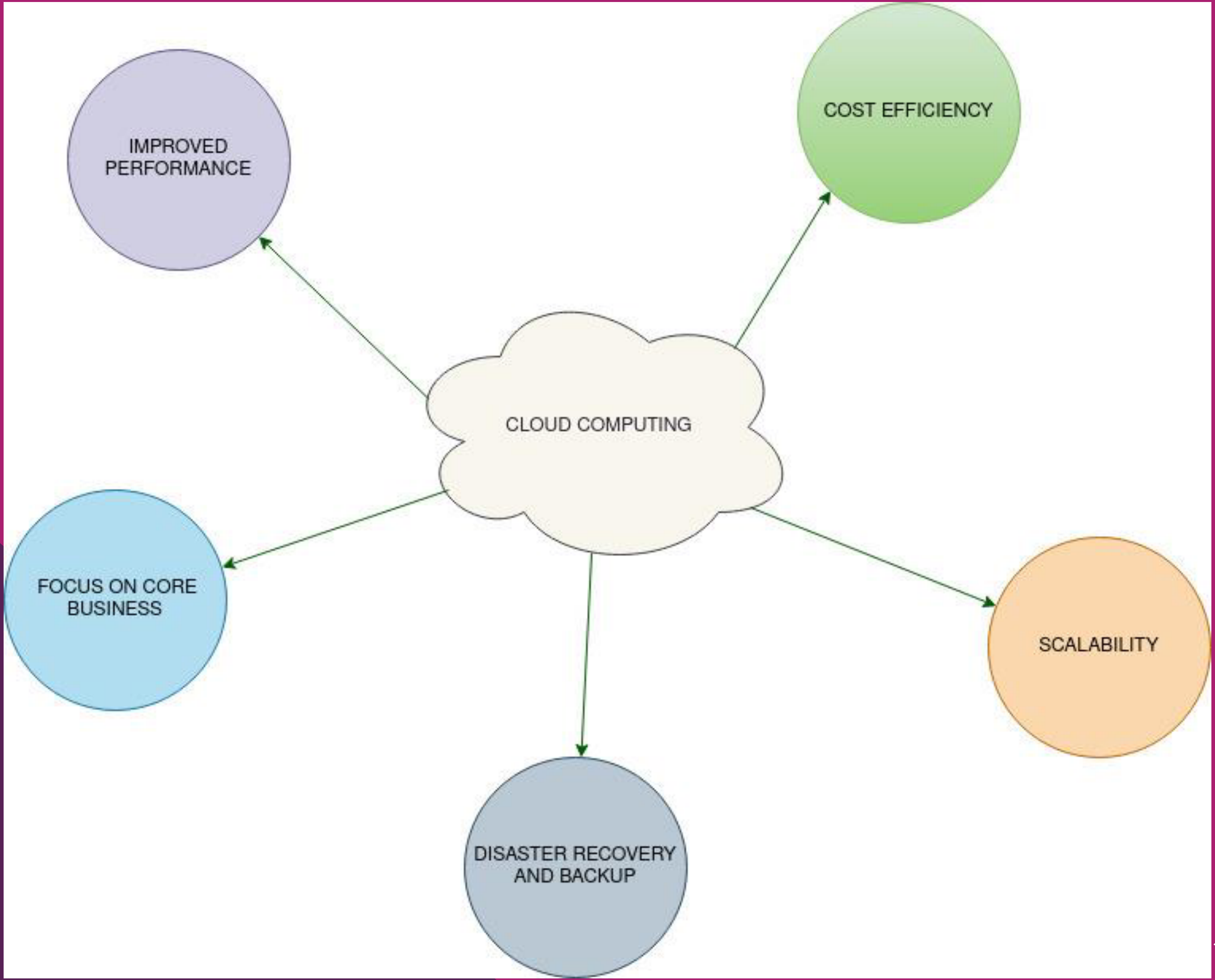
HOW CLOUD COMPUTING HELP BUSINESS?



Cloud Computing

Cloud computing is the on-demand delivery of IT resources over the Internet with pay-as-you-go pricing. Instead of buying, owning, and maintaining physical data centers and servers, you can access technology services, such as computing power, storage, and databases, on an as-needed basis from a cloud provider like Amazon Web Services (AWS).

Visual Diagram

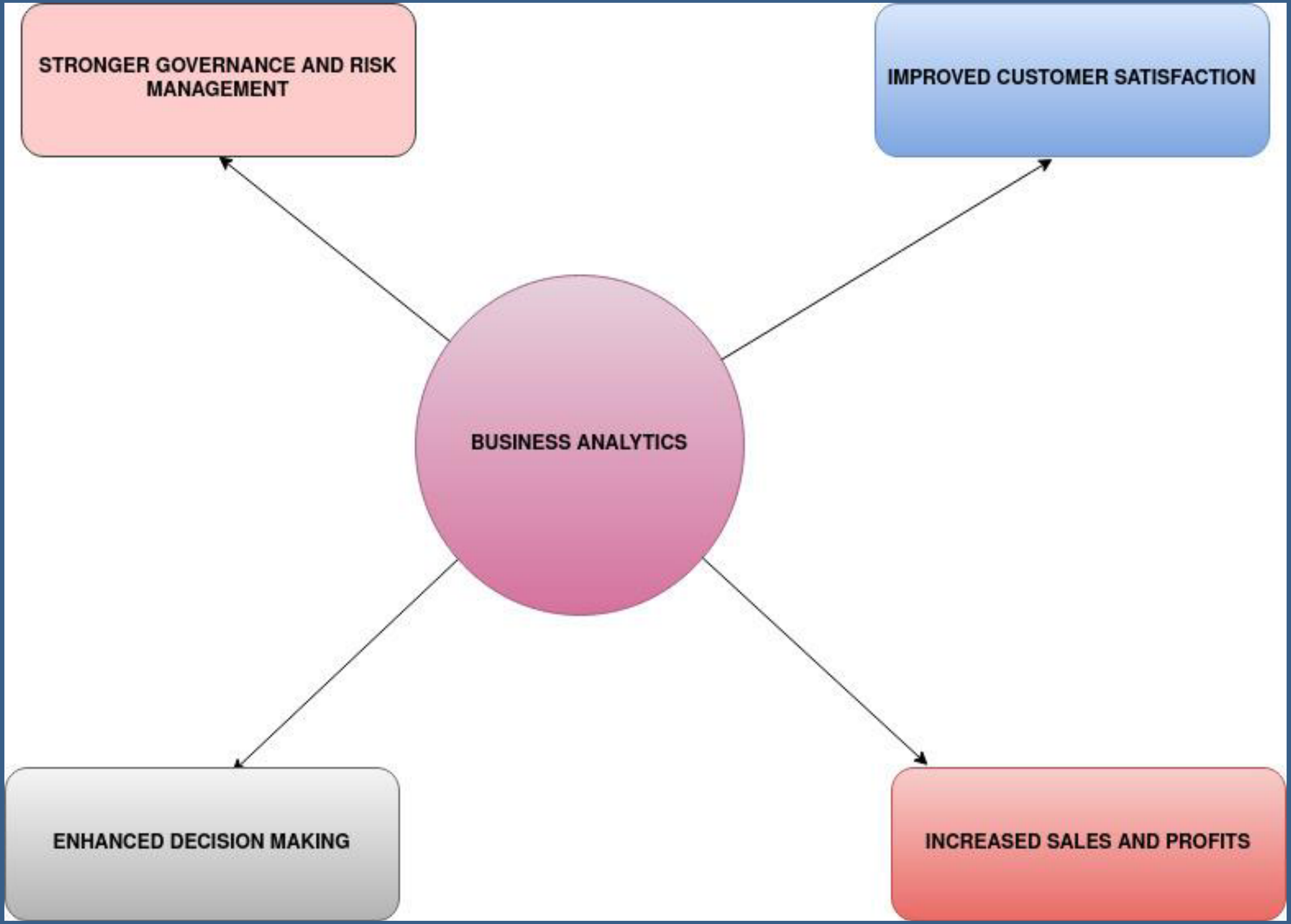


HOW ANALYTICS HELP BUSINESSES ?

Business Analytics

Business analytics involves the systematic analysis of data to inform business decisions and strategies. It combines statistical analysis, data mining, predictive modeling, and data visualization to derive actionable insights that can improve performance, drive growth, and enhance decision-making processes.

Visual Diagram



BUSINESS ANALYTICS

STRONGER GOVERNANCE AND RISK MANAGEMENT

IMPROVED CUSTOMER SATISFACTION

ENHANCED DECISION MAKING

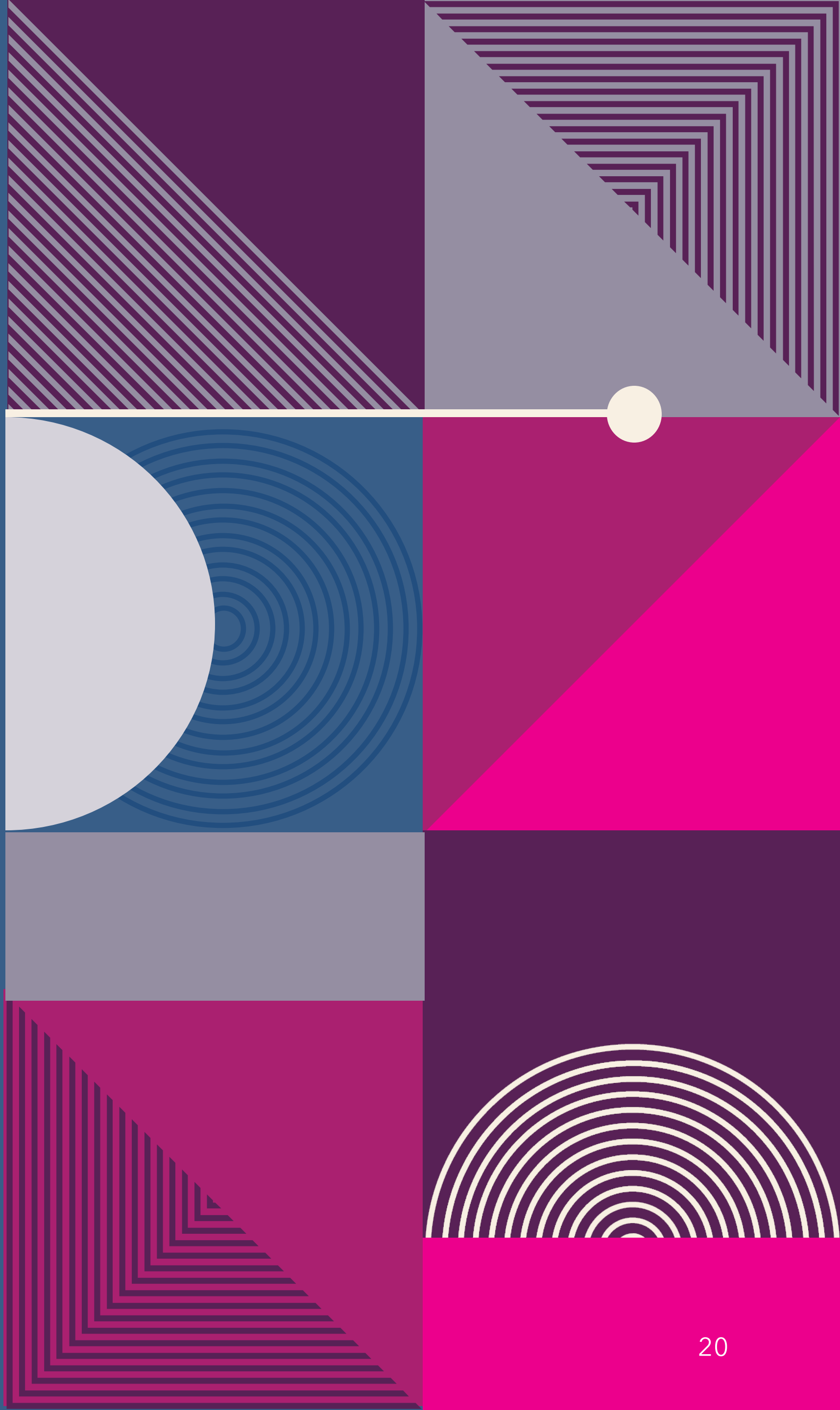
INCREASED SALES AND PROFITS

OWASP TOP 10

1. The Open Web Application Security Project (OWASP) is a non-profit organization dedicated to improving software security. Founded in 2001 by Mark Curphey, OWASP is an open community that encourages the informed use of application security technologies
2. The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.

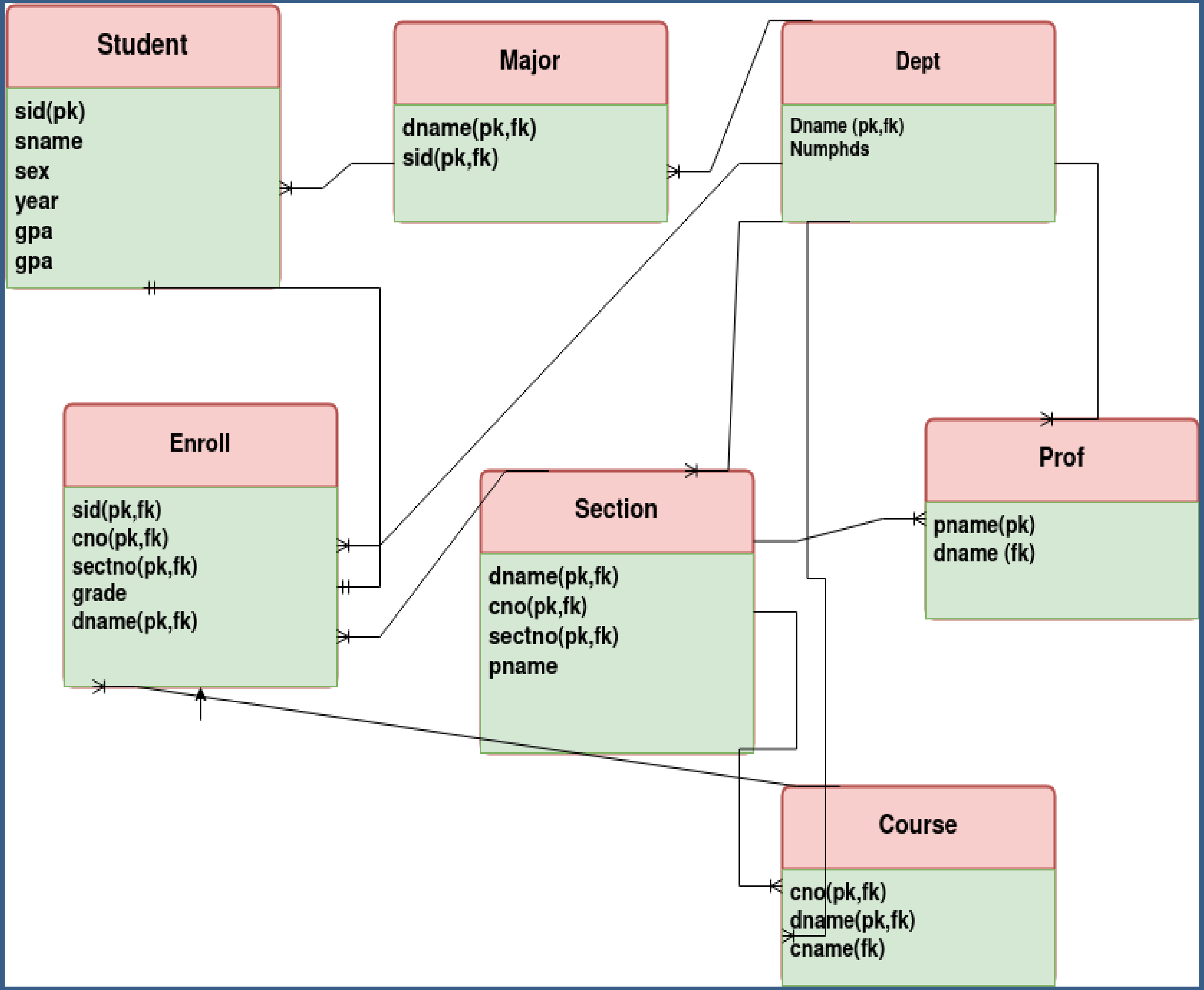
TOP 10 WEB APPLICATION SECURITY RISKS

1. A01:2021-Broken Access Control
2. A02:2021-Cryptographic Failures
3. A03:2021-Injection
4. A04:2021-Insecure Design
5. A05:2021-Security Misconfiguration
6. A06:2021-Vulnerable and Outdated Components
7. A07:2021-Identification and Authentication Failures
8. A08:2021-Software and Data Integrity Failures
9. A09:2021-Security Logging and Monitoring Failures
10. A10:2021-Server-Side Request Forgery



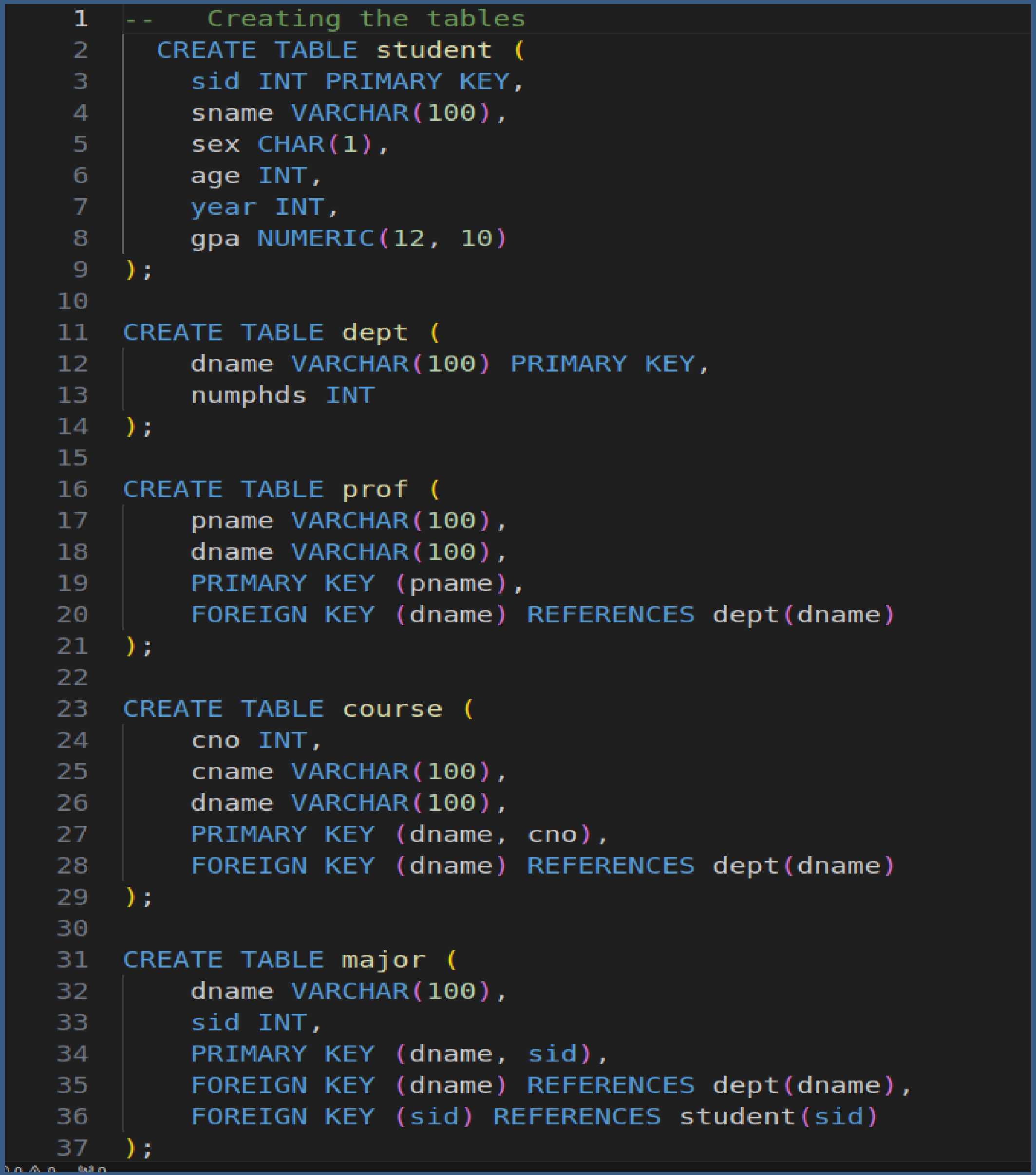
SQL ANALYSIS TASK

ER DIAGRAM



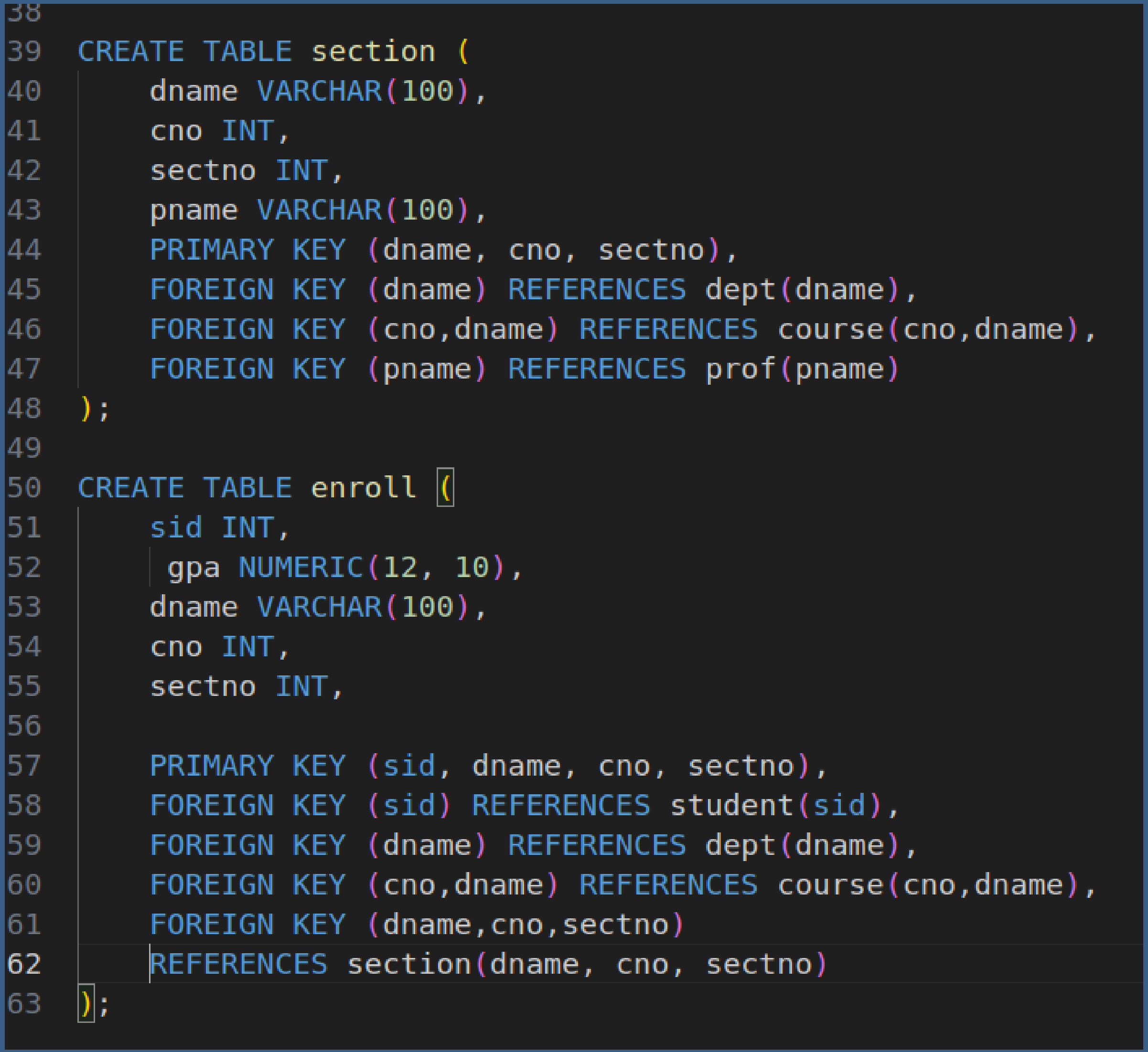
Student
sid(pk)
sname
sex
year
gpa
gpa
Major
dname(pk,fk)
sid(pk,fk)
Dept
Dname (pk,fk)
Numphds
Enroll
sid(pk,fk)
cno(pk,fk)
sectno(pk,fk)
grade
dname(pk,fk)
Section
dname(pk,fk)
cno(pk,fk)
sectno(pk,fk)
pname
Prof
pname(pk)
dname (fk)
Course
cno(pk,fk)
dname(pk,fk)
cname(fk)

CREATING THE TABLES



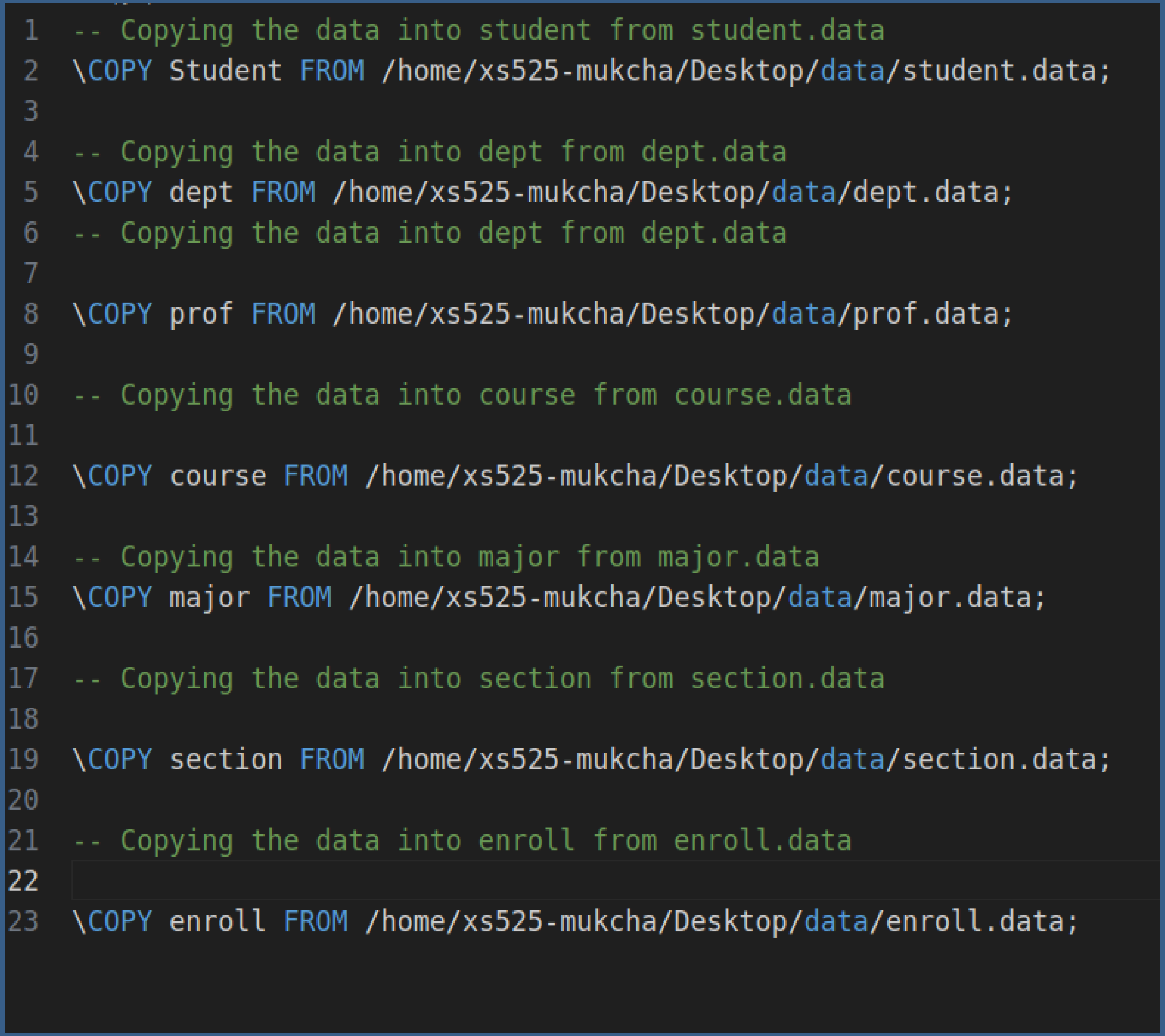
```
--   Creating the tables
  CREATE TABLE student (
    sid INT PRIMARY KEY,
    sname VARCHAR(100),
    sex CHAR(1),
    age INT,
    year INT,
    gpa NUMERIC(12, 10)
);

CREATE TABLE dept (
    dname VARCHAR(100) PRIMARY KEY,
    numphds INT
);

CREATE TABLE prof (
    pname VARCHAR(100),
    dname VARCHAR(100),
    PRIMARY KEY (pname),
    FOREIGN KEY (dname) REFERENCES dept(dname)
);

CREATE TABLE course (
    cno INT,
    cname VARCHAR(100),
    dname VARCHAR(100),
    PRIMARY KEY (dname, cno),
    FOREIGN KEY (dname) REFERENCES dept(dname)
);

CREATE TABLE major (
    dname VARCHAR(100),
    sid INT,
    PRIMARY KEY (dname, sid),
    FOREIGN KEY (dname) REFERENCES dept(dname),
    FOREIGN KEY (sid) REFERENCES student(sid)
);
```

CREATING THE TABLES

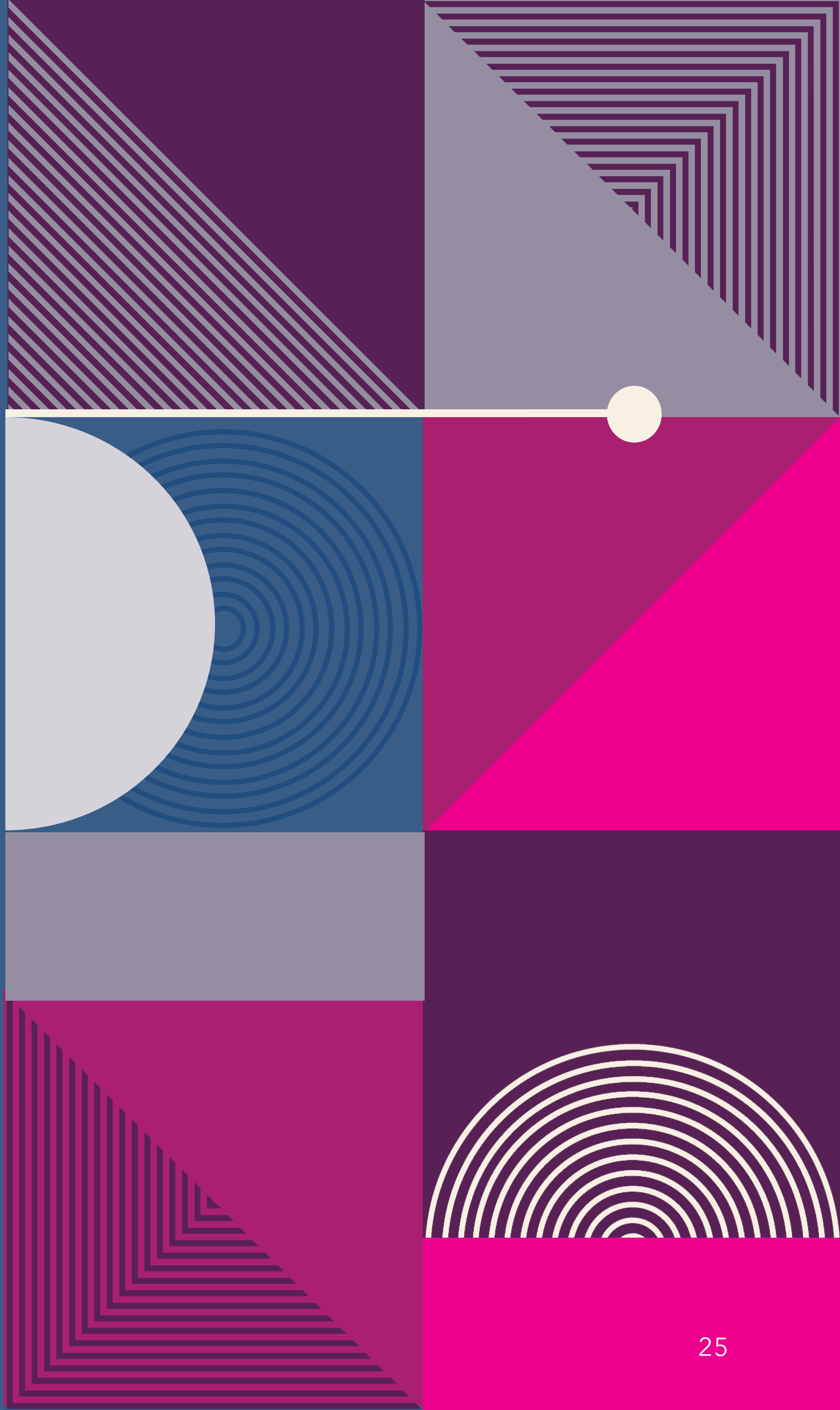


```sql

CREATE TABLE section (
    dname VARCHAR(100),
    cno INT,
    sectno INT,
    pname VARCHAR(100),
    PRIMARY KEY (dname, cno, sectno),
    FOREIGN KEY (dname) REFERENCES dept(dname),
    FOREIGN KEY (cno,dname) REFERENCES course(cno,dname),
    FOREIGN KEY (pname) REFERENCES prof(pname)
);

CREATE TABLE enroll (
    sid INT,
     gpa NUMERIC(12, 10),
    dname VARCHAR(100),
    cno INT,
    sectno INT,

    PRIMARY KEY (sid, dname, cno, sectno),
    FOREIGN KEY (sid) REFERENCES student(sid),
    FOREIGN KEY (dname) REFERENCES dept(dname),
    FOREIGN KEY (cno,dname) REFERENCES course(cno,dname),
    FOREIGN KEY (dname,cno,sectno)
    REFERENCES section(dname, cno, sectno)
);
```

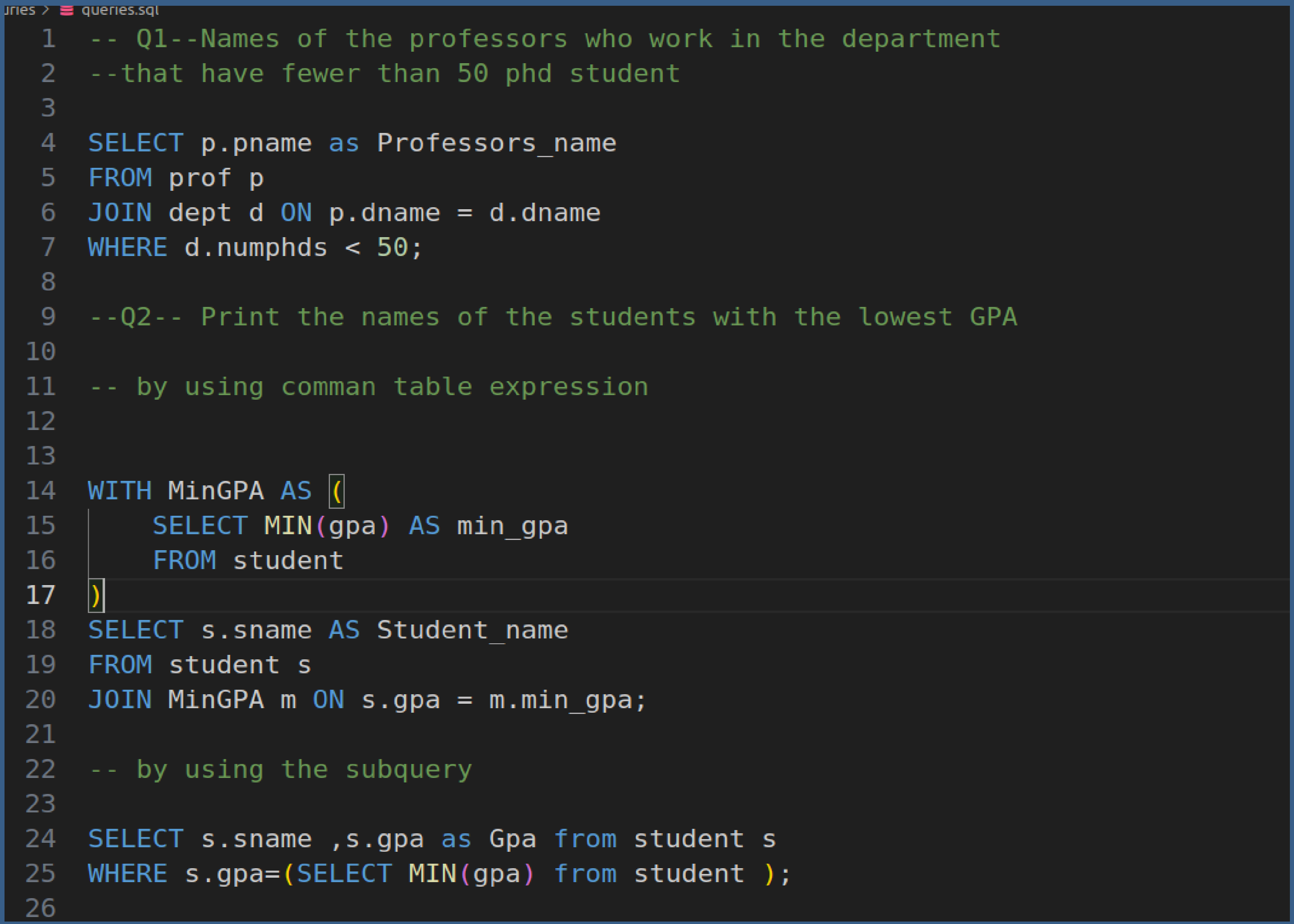
DATA COPYING



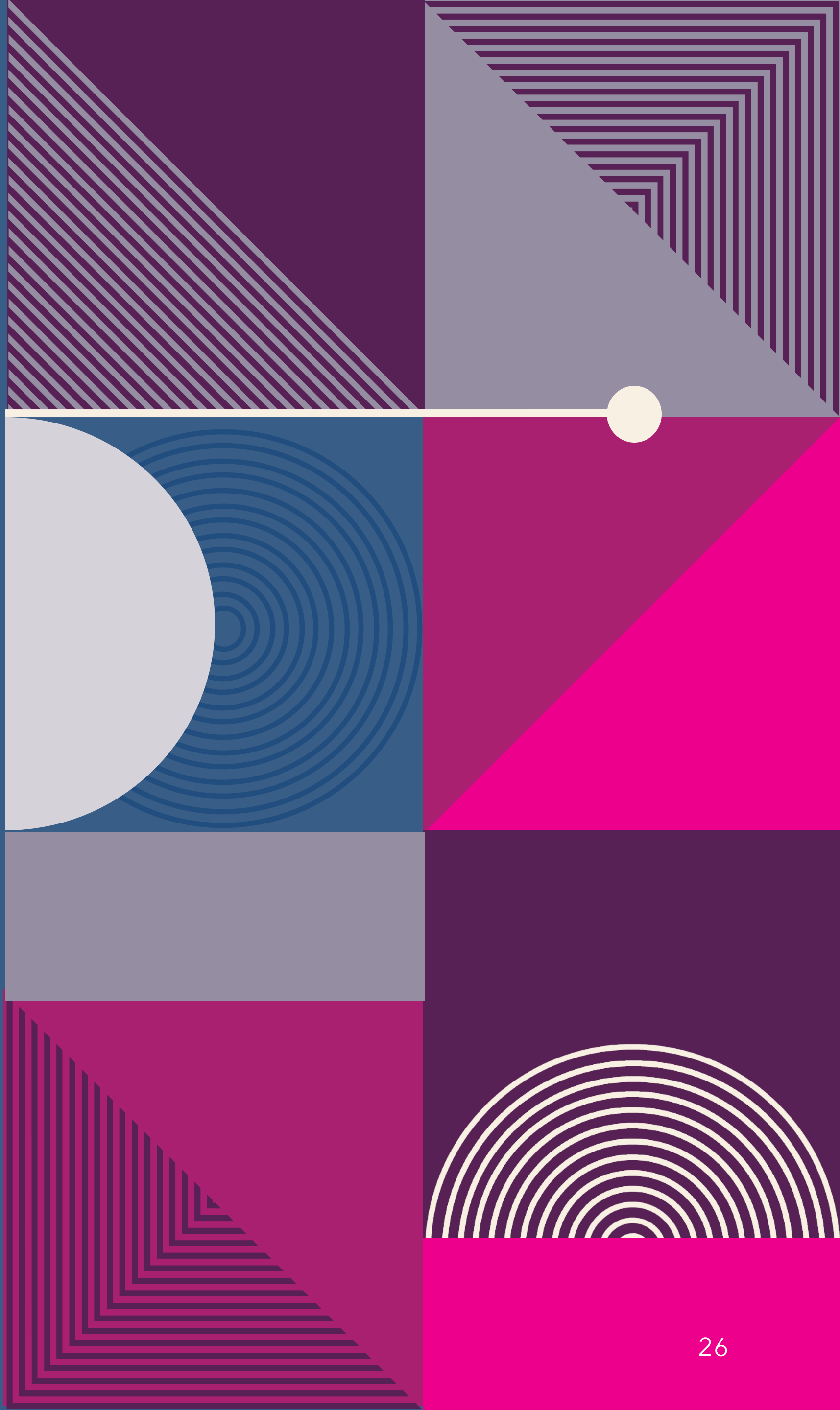
```
-- Copying the data into student from student.data
\COPY Student FROM /home/xs525-mukcha/Desktop/data/student.data;

-- Copying the data into dept from dept.data
\COPY dept FROM /home/xs525-mukcha/Desktop/data/dept.data;
-- Copying the data into dept from dept.data

\COPY prof FROM /home/xs525-mukcha/Desktop/data/prof.data;

-- Copying the data into course from course.data

\COPY course FROM /home/xs525-mukcha/Desktop/data/course.data;

-- Copying the data into major from major.data
\COPY major FROM /home/xs525-mukcha/Desktop/data/major.data;

-- Copying the data into section from section.data

\COPY section FROM /home/xs525-mukcha/Desktop/data/section.data;

-- Copying the data into enroll from enroll.data

\COPY enroll FROM /home/xs525-mukcha/Desktop/data/enroll.data;
```



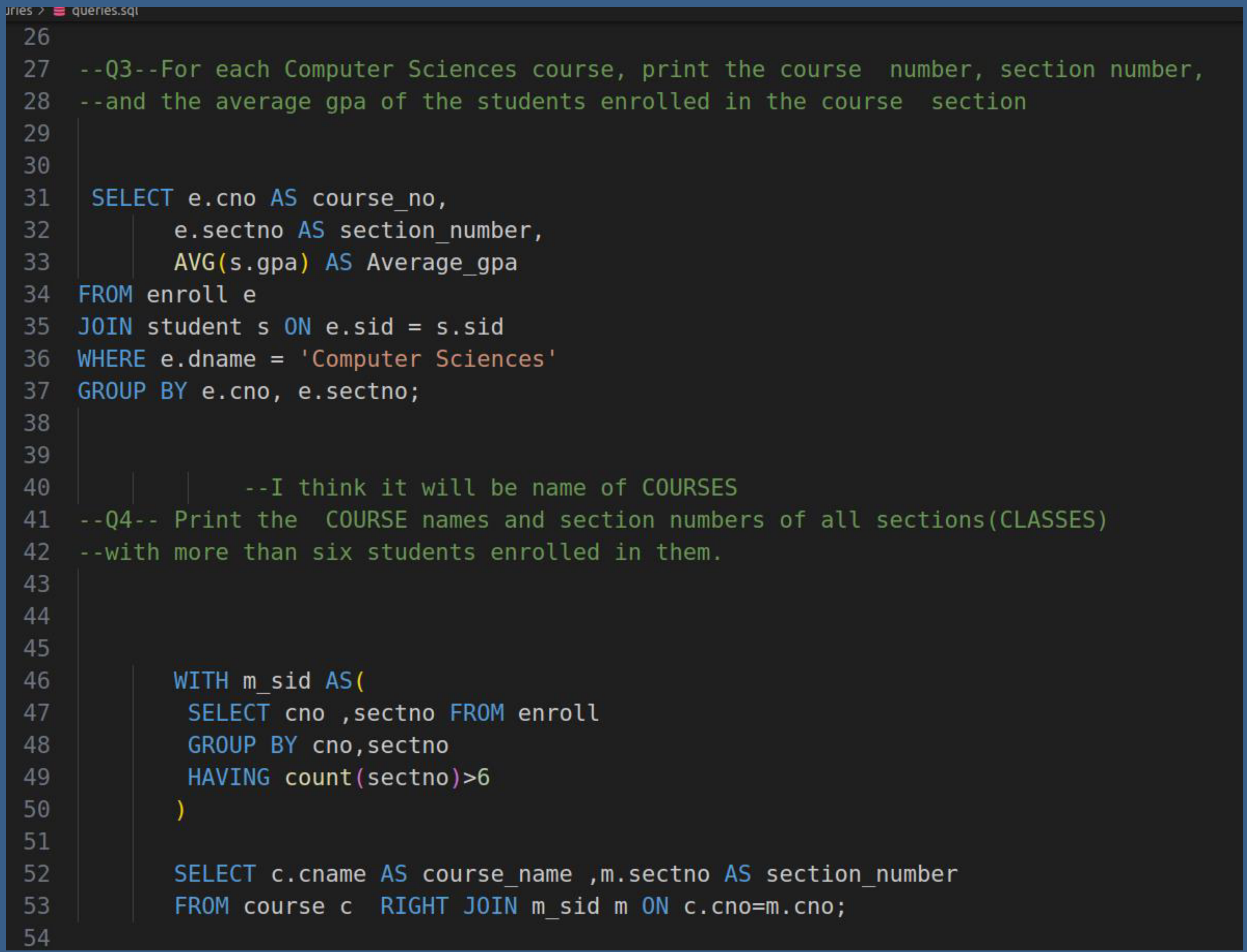
QUESTION 1 AND QUESTION2



```sql
-- Q1--Names of the professors who work in the department
--that have fewer than 50 phd student

SELECT p.pname as Professors_name
FROM prof p
JOIN dept d ON p.dname = d.dname
WHERE d.numphds < 50;

--Q2-- Print the names of the students with the lowest GPA

-- by using comman table expression


WITH MinGPA AS (
    SELECT MIN(gpa) AS min_gpa
    FROM student
)
SELECT s.sname AS Student_name
FROM student s
JOIN MinGPA m ON s.gpa = m.min_gpa;

-- by using the subquery

SELECT s.sname ,s.gpa as Gpa from student s
WHERE s.gpa=(SELECT MIN(gpa) from student );

```

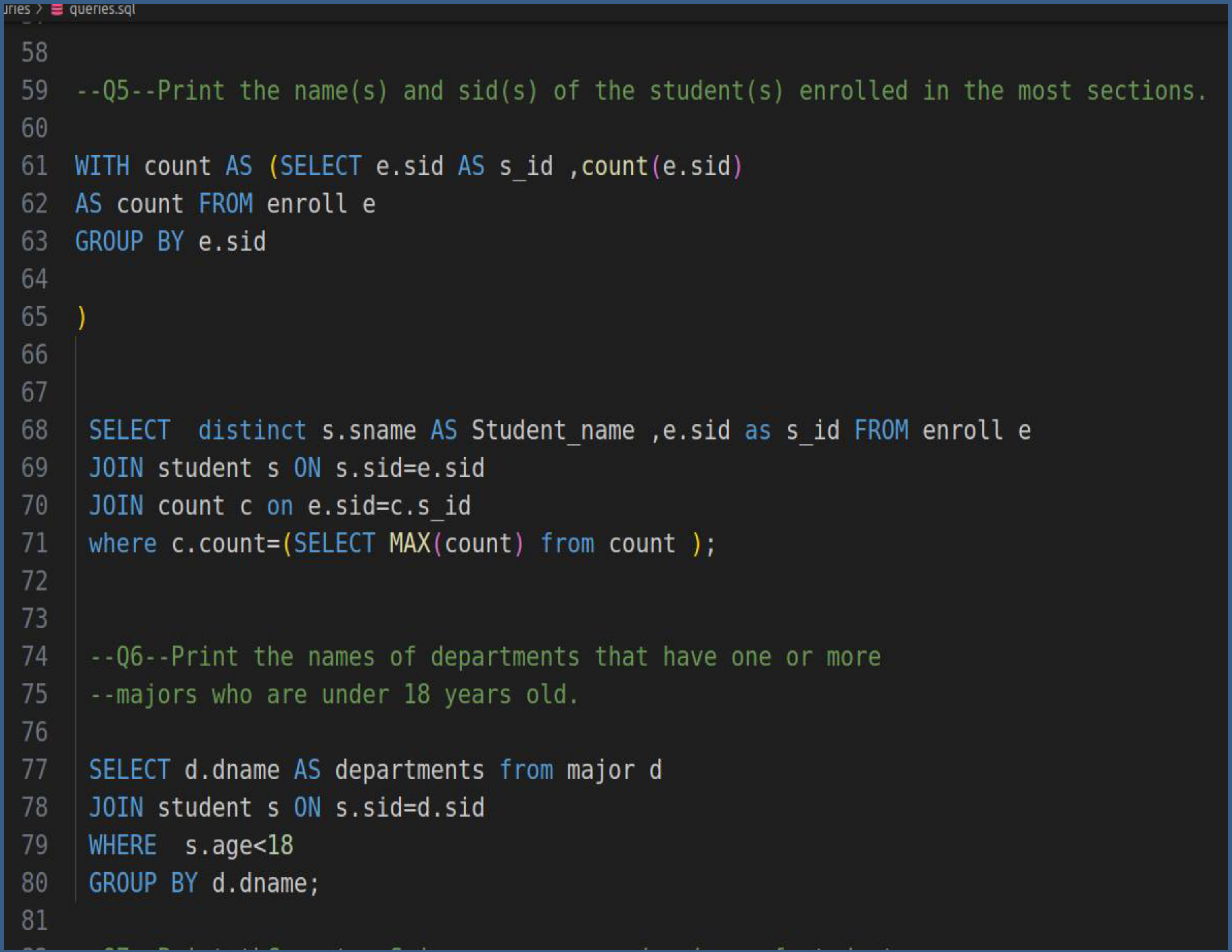


QUESTION 3 AND QUESTION4



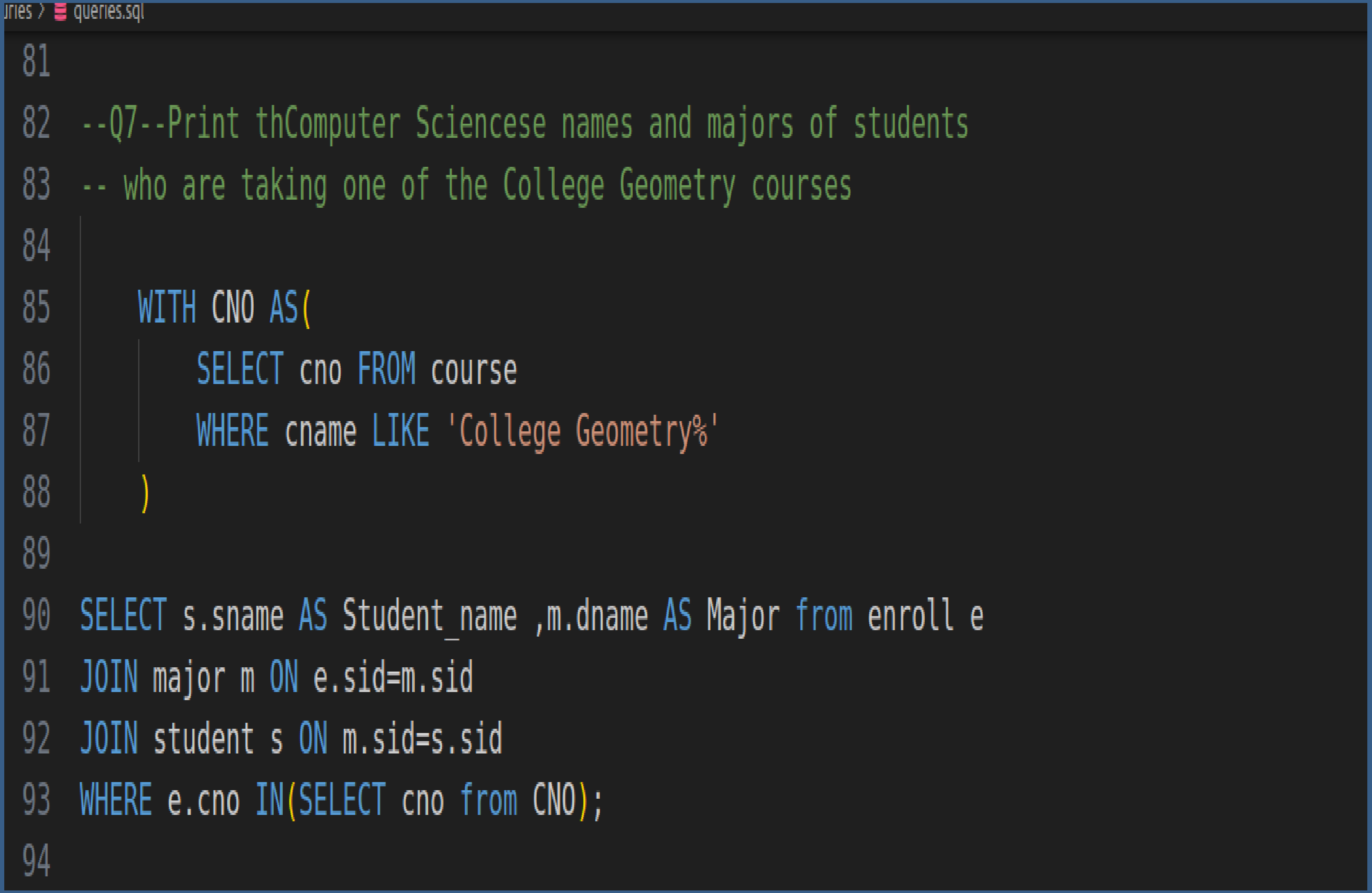
```sql

--Q3--For each Computer Sciences course, print the course  number, section number,
--and the average gpa of the students enrolled in the course  section


 SELECT e.cno AS course_no,
       e.sectno AS section_number,
       AVG(s.gpa) AS Average_gpa
FROM enroll e
JOIN student s ON e.sid = s.sid
WHERE e.dname = 'Computer Sciences'
GROUP BY e.cno, e.sectno;


            --I think it will be name of COURSES
--Q4-- Print the  COURSE names and section numbers of all sections(CLASSES)
--with more than six students enrolled in them.



       WITH m_sid AS(
        SELECT cno ,sectno FROM enroll
        GROUP BY cno,sectno
        HAVING count(sectno)>6
       )

       SELECT c.cname AS course_name ,m.sectno AS section_number
       FROM course c  RIGHT JOIN m_sid m ON c.cno=m.cno;

```

QUESTION 5 AND QUESTION 6



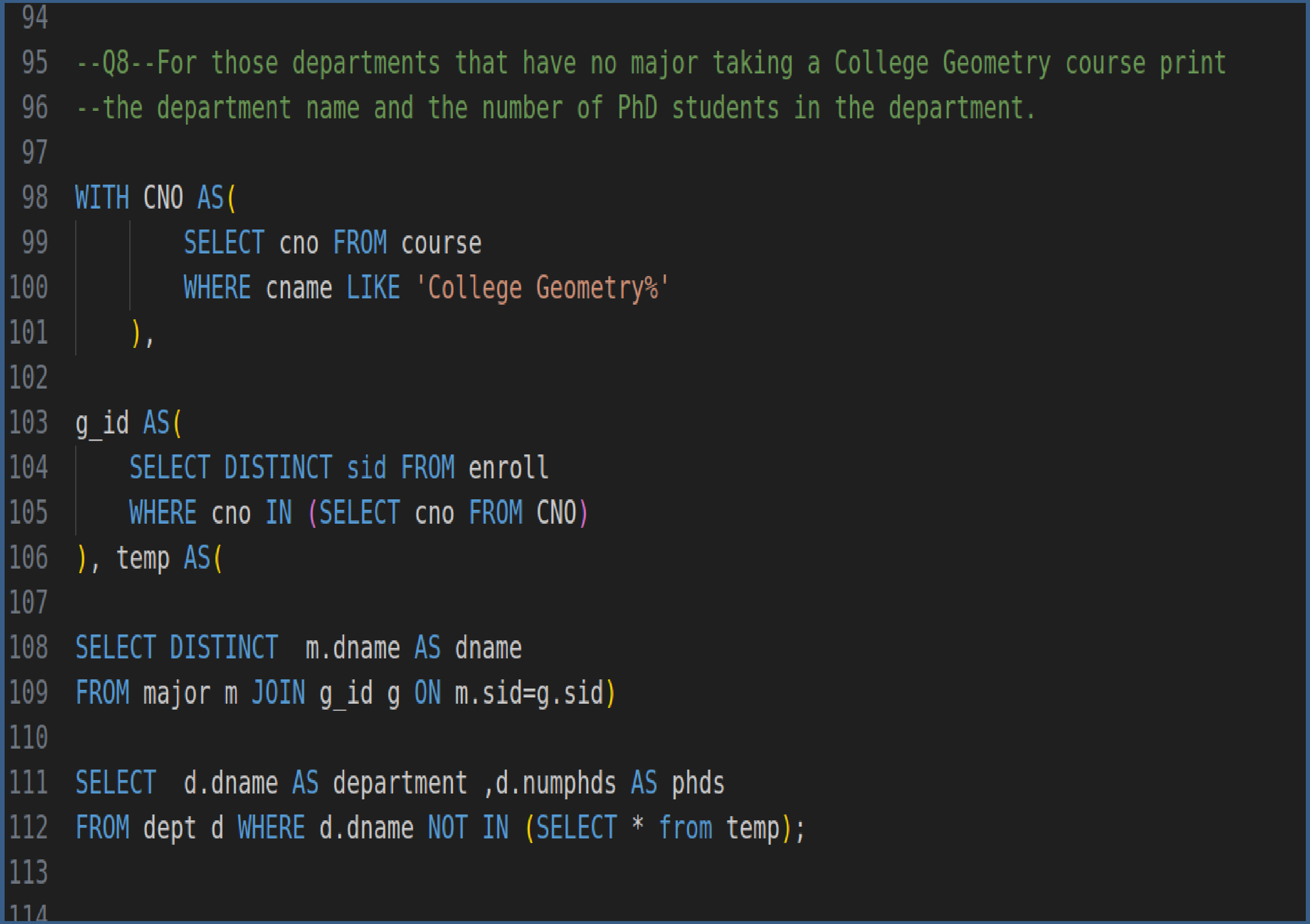
```
--Q5--Print the name(s) and sid(s) of the student(s) enrolled in the most sections.

WITH count AS (SELECT e.sid AS s_id ,count(e.sid)
AS count FROM enroll e
GROUP BY e.sid

)


 SELECT  distinct s.sname AS Student_name ,e.sid as s_id FROM enroll e
 JOIN student s ON s.sid=e.sid
 JOIN count c on e.sid=c.s_id
 where c.count=(SELECT MAX(count) from count );


 --Q6--Print the names of departments that have one or more
 --majors who are under 18 years old.

 SELECT d.dname AS departments from major d
 JOIN student s ON s.sid=d.sid
 WHERE  s.age<18
 GROUP BY d.dname;

```

QUESTION 7



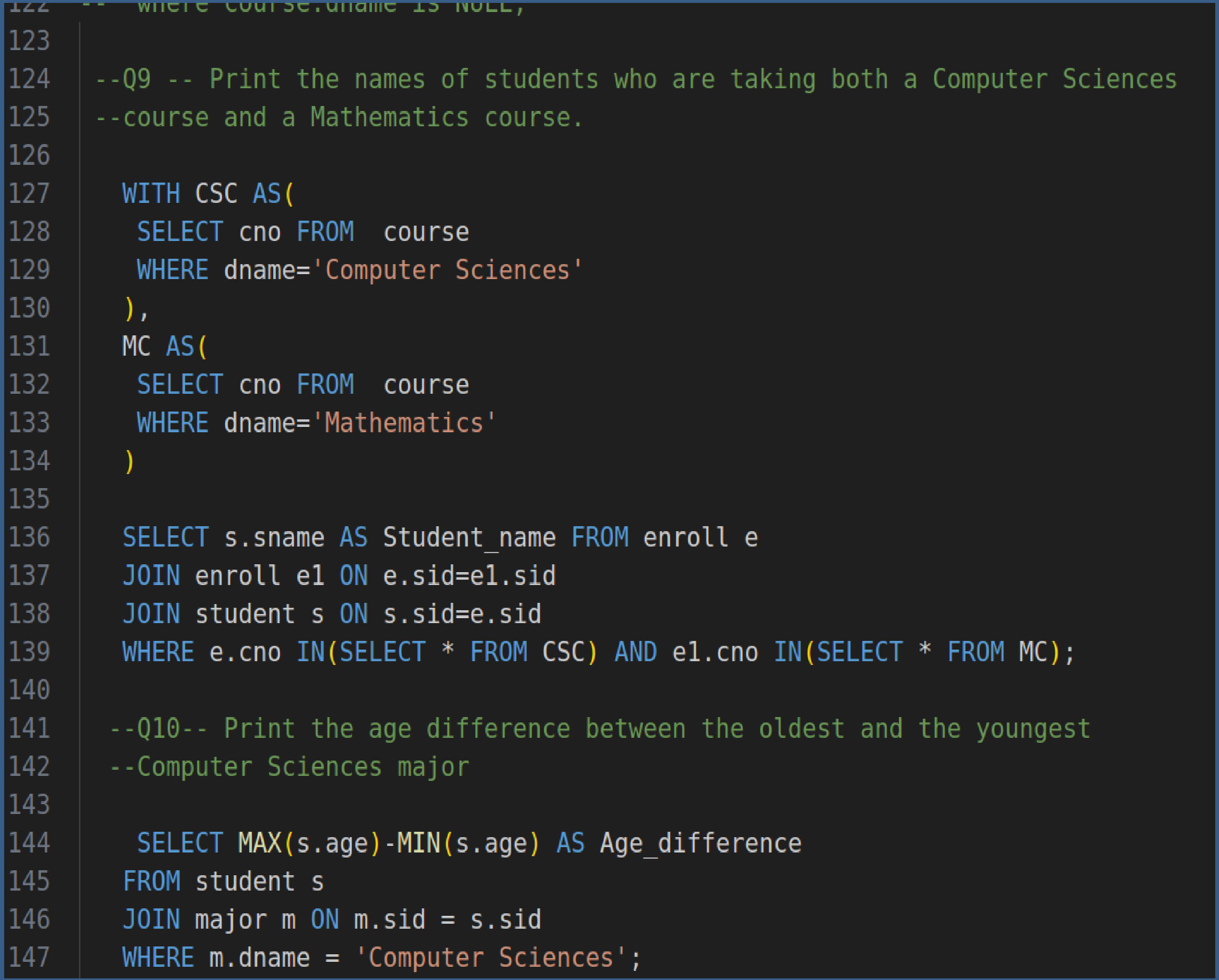
```sql

--Q7--Print thComputer Sciencese names and majors of students
-- who are taking one of the College Geometry courses

    WITH CNO AS(
        SELECT cno FROM course
        WHERE cname LIKE 'College Geometry%'
    )

SELECT s.sname AS Student_name ,m.dname AS Major from enroll e
JOIN major m ON e.sid=m.sid
JOIN student s ON m.sid=s.sid
WHERE e.cno IN(SELECT cno from CNO);

```

QUESTION 8



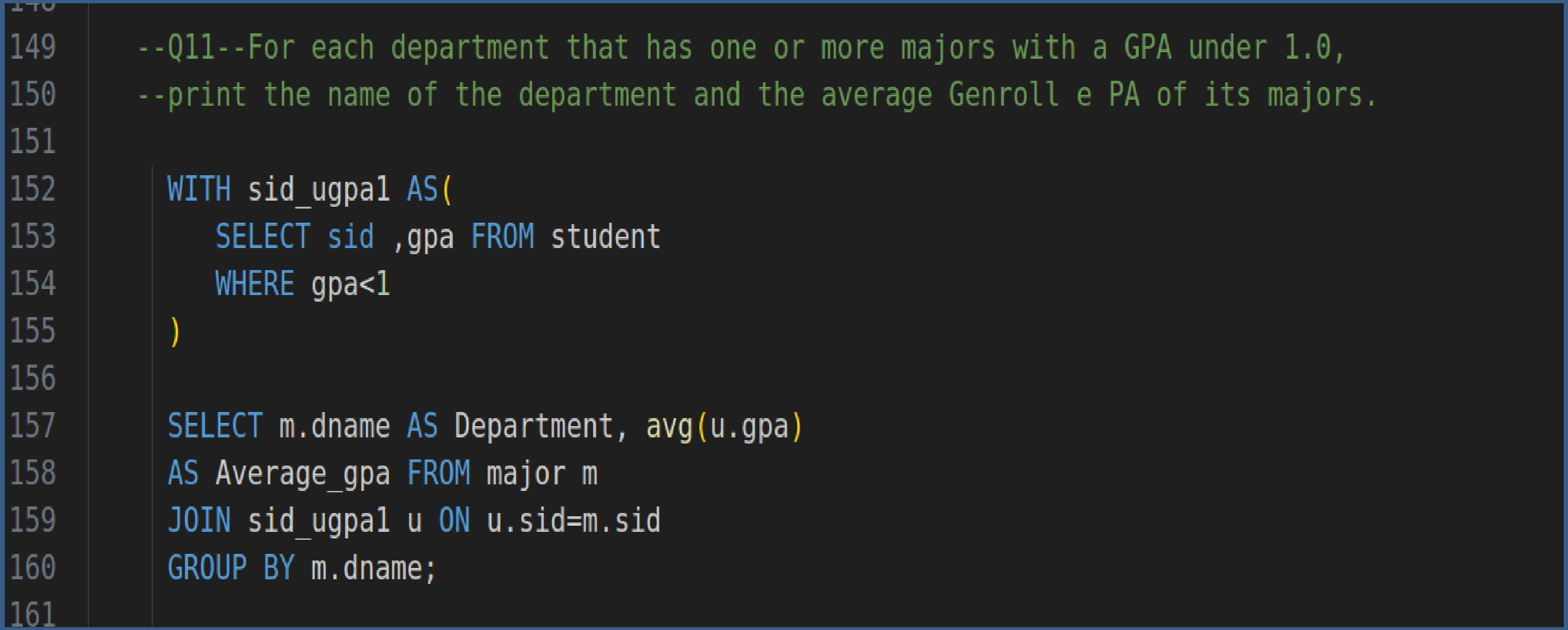
```sql

--Q8--For those departments that have no major taking a College Geometry course print
--the department name and the number of PhD students in the department.

WITH CNO AS(
        SELECT cno FROM course
        WHERE cname LIKE 'College Geometry%'
    ),

g_id AS(
    SELECT DISTINCT sid FROM enroll
    WHERE cno IN (SELECT cno FROM CNO)
), temp AS(

SELECT DISTINCT  m.dname AS dname
FROM major m JOIN g_id g ON m.sid=g.sid)

SELECT  d.dname AS department ,d.numphds AS phds
FROM dept d WHERE d.dname NOT IN (SELECT * from temp);

```

QUESTION 9AND 10



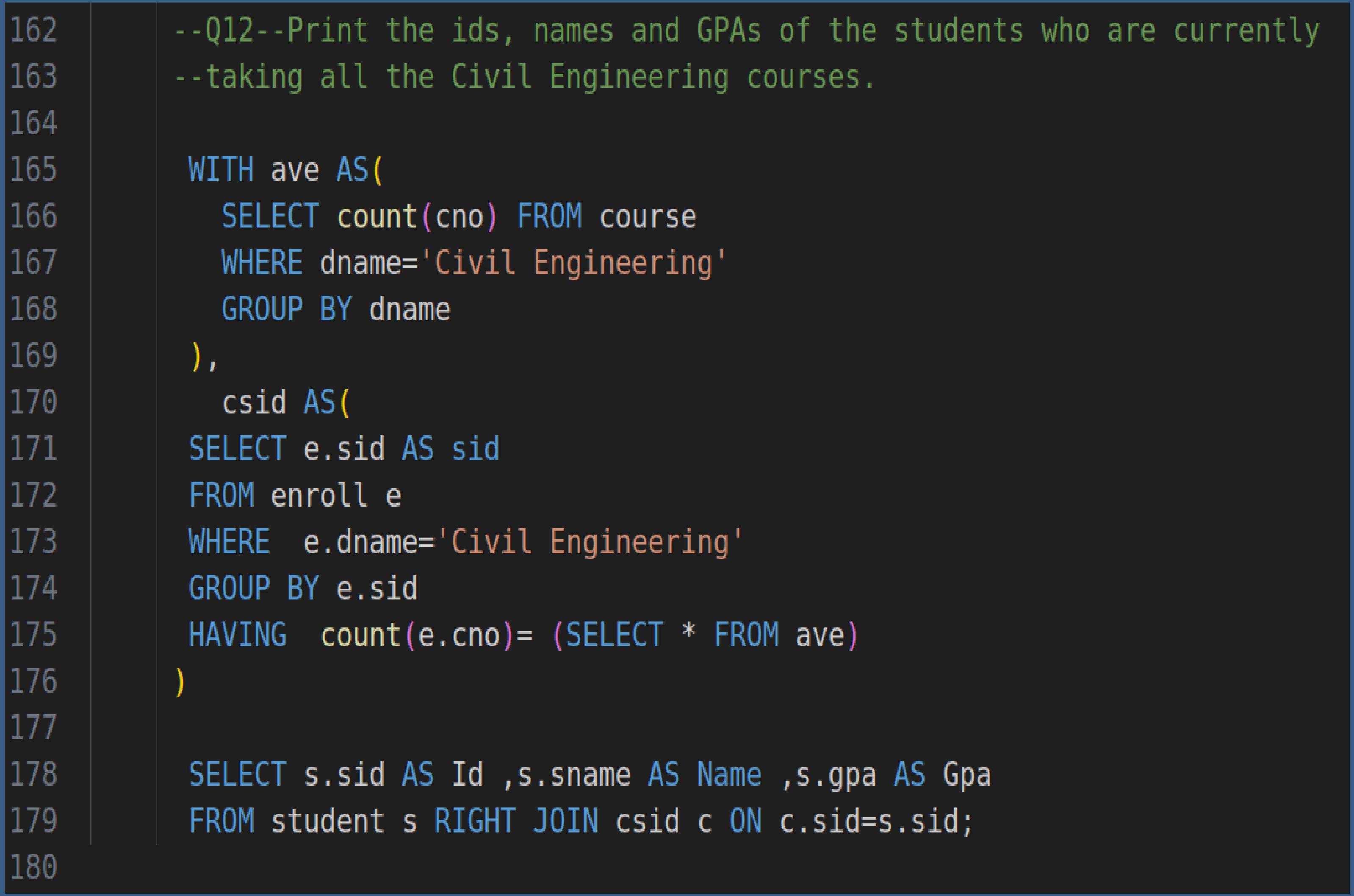
```
--  where course.dname is NULL;

 --Q9 -- Print the names of students who are taking both a Computer Sciences
 --course and a Mathematics course.

   WITH CSC AS(
    SELECT cno FROM  course
    WHERE dname='Computer Sciences'
   ),
   MC AS(
    SELECT cno FROM  course
    WHERE dname='Mathematics'
   )

   SELECT s.sname AS Student_name FROM enroll e
   JOIN enroll e1 ON e.sid=e1.sid
   JOIN student s ON s.sid=e.sid
   WHERE e.cno IN(SELECT * FROM CSC) AND e1.cno IN(SELECT * FROM MC);

  --Q10-- Print the age difference between the oldest and the youngest
  --Computer Sciences major

    SELECT MAX(s.age)-MIN(s.age) AS Age_difference
   FROM student s
   JOIN major m ON m.sid = s.sid
   WHERE m.dname = 'Computer Sciences';
```

QUESTION 11



```sql
--Q11--For each department that has one or more majors with a GPA under 1.0,
--print the name of the department and the average Genroll e PA of its majors.

  WITH sid_ugpa1 AS(
     SELECT sid ,gpa FROM student
     WHERE gpa<1
  )

  SELECT m.dname AS Department, avg(u.gpa)
  AS Average_gpa FROM major m
  JOIN sid_ugpa1 u ON u.sid=m.sid
  GROUP BY m.dname;
```

QUESTION 12

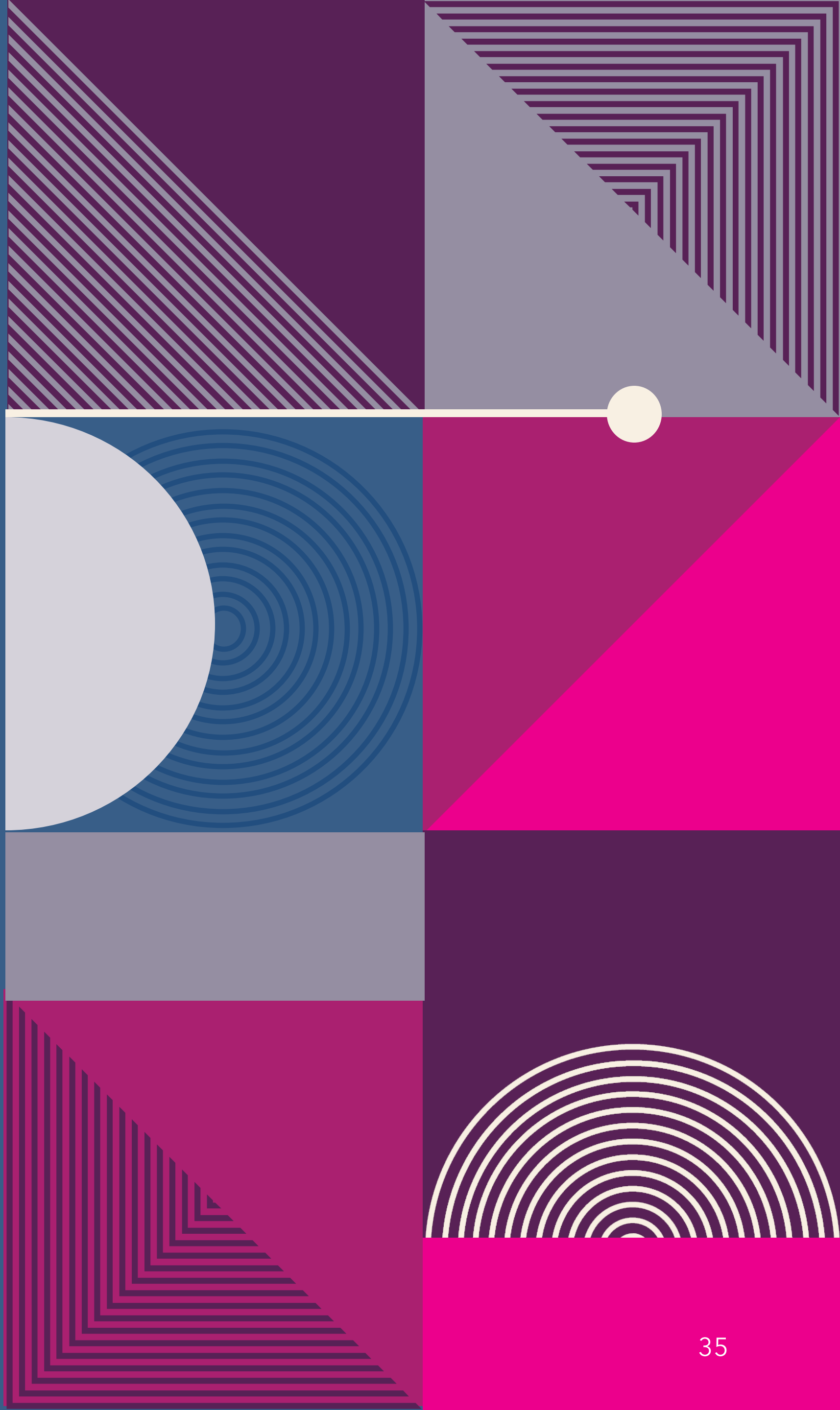


```
--Q12--Print the ids, names and GPAs of the students who are currently
--taking all the Civil Engineering courses.

 WITH ave AS(
   SELECT count(cno) FROM course
   WHERE dname='Civil Engineering'
   GROUP BY dname
 ),
   csid AS(
 SELECT e.sid AS sid
 FROM enroll e
 WHERE  e.dname='Civil Engineering'
 GROUP BY e.sid
 HAVING  count(e.cno)= (SELECT * FROM ave)
)

 SELECT s.sid AS Id ,s.sname AS Name ,s.gpa AS Gpa
 FROM student s RIGHT JOIN csid c ON c.sid=s.sid;
```

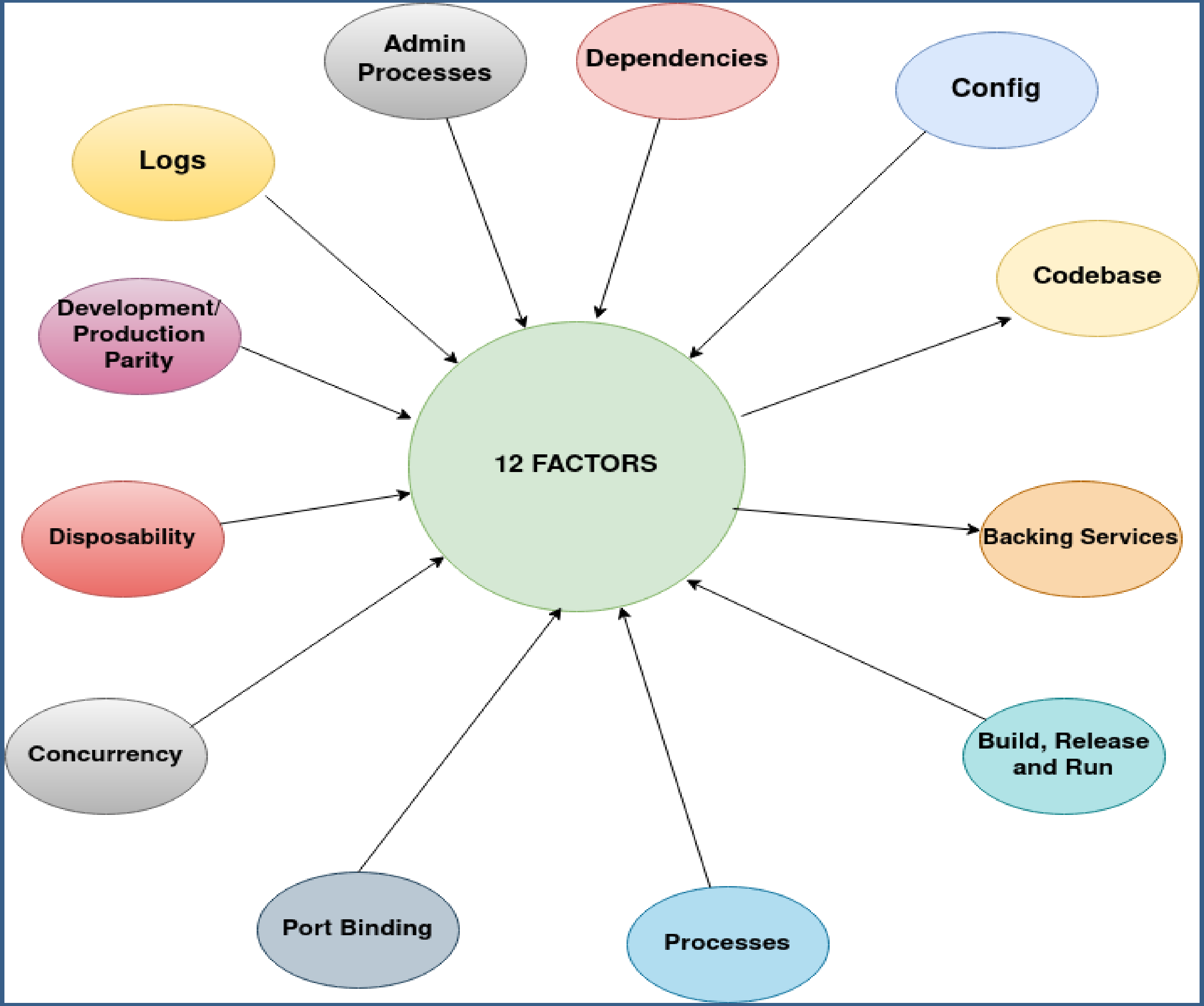
THE TWELE FACTOR WEB APP

WHAT IS TWELVE-FACTOR APP?

The Twelve-factor app is a set of 12 principles or best practices for building web applications which now days are more commonly known as Software-As-A-Service (SAAS) applications. It was published by the co-founder of Heroku, Mr. Adam Wiggins in 2011



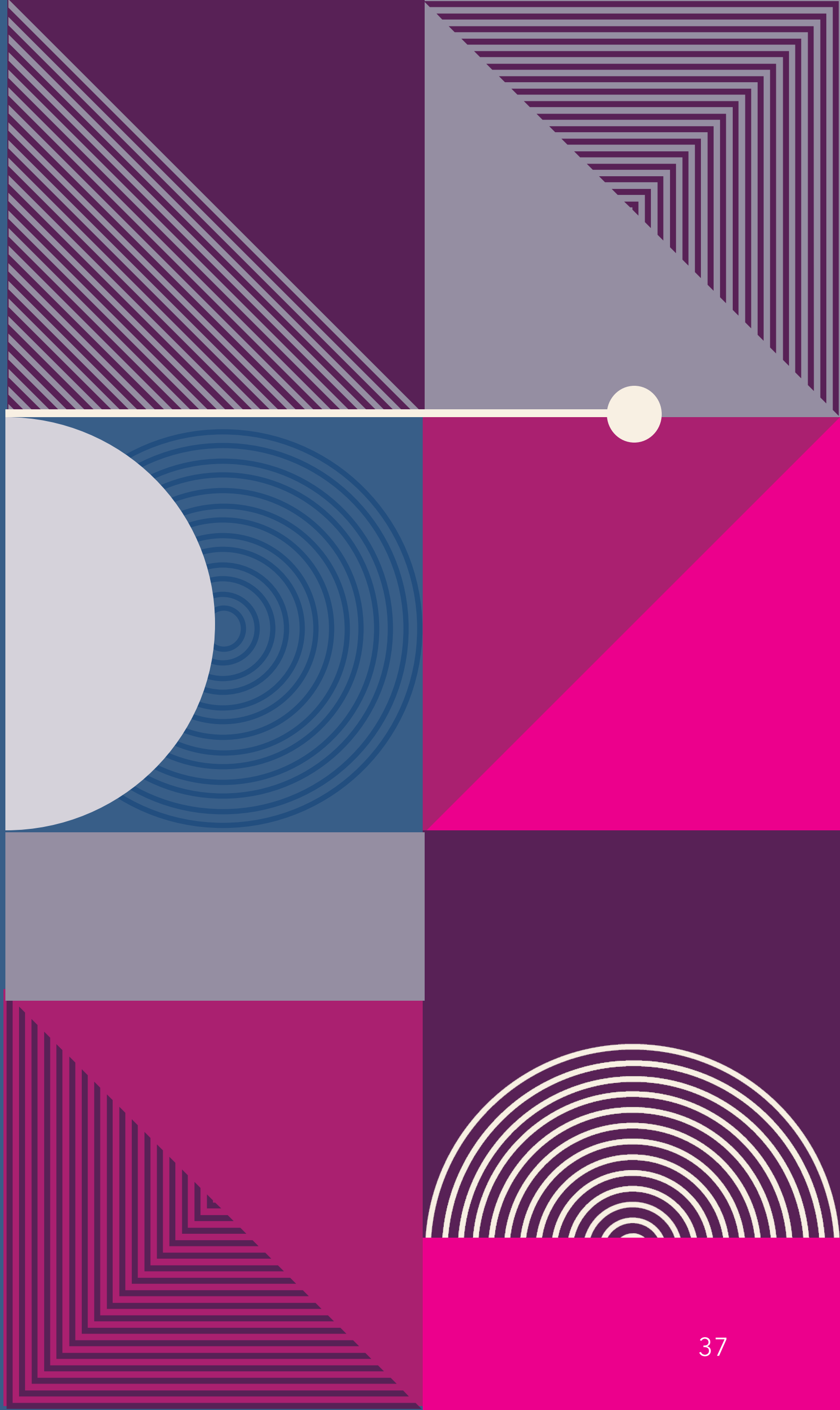
12 FACTORS


Admin Processes
Dependencies
Config
Logs
Codebase
Development/ Production Parity
12 FACTORS
Disposability
Backing Services
Concurrency
Build, Release and Run
Port Binding
Processes

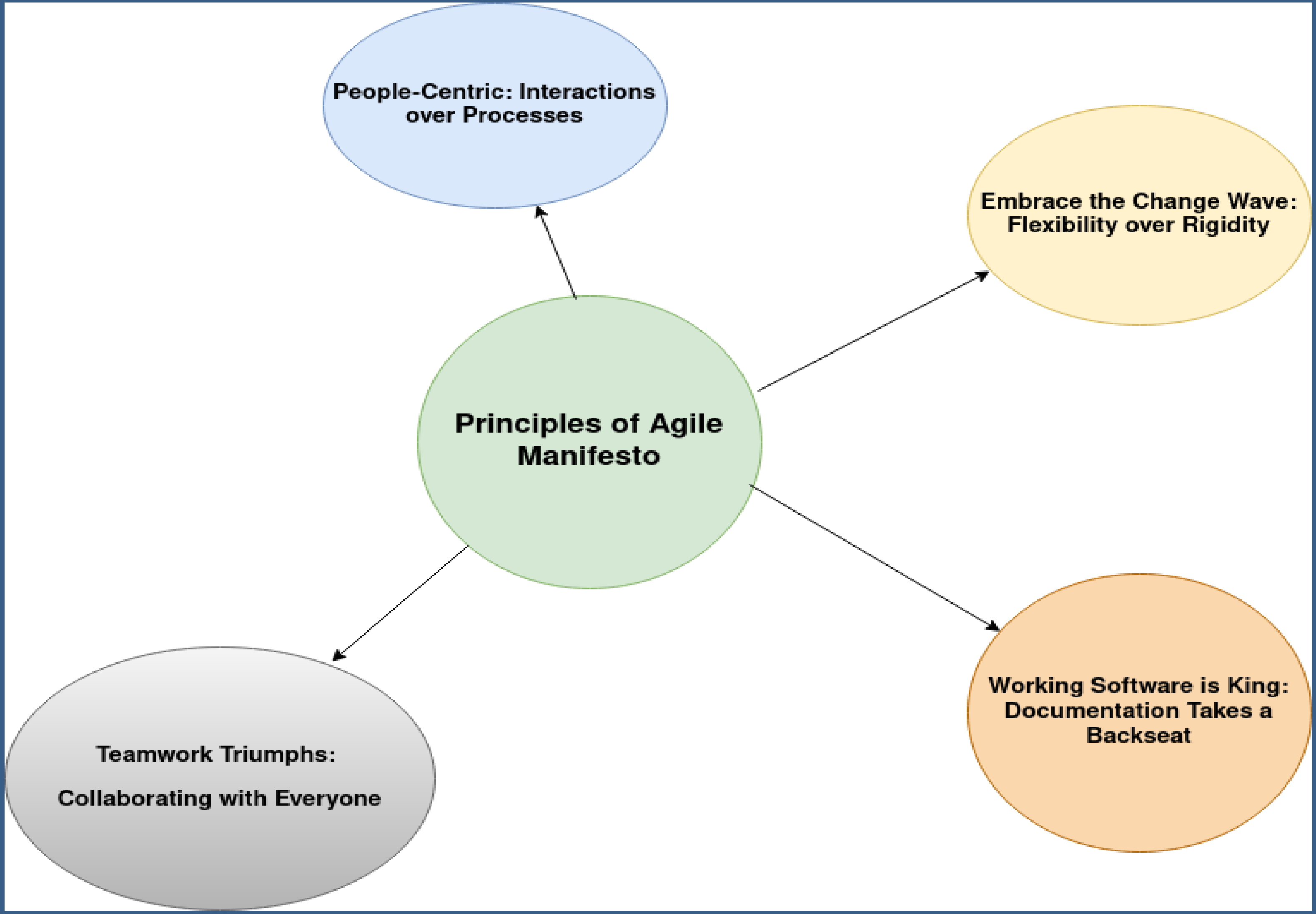
AGILE MANIFESTO

The Agile Manifesto in software development means a set of fundamental documents that solely outline a series of values and principles that focus on the Agile methodologies in software development.

1. This manifesto underscores the significance of flexibility, collaboration, and customer contentment.
2. It places importance on individuals and interactions above processes and tools, working software over extensive documentation.



PRINCIPLES OF AGILE MANIFESTO



Principles of Agile Manifesto

People-Centric: Interactions over Processes

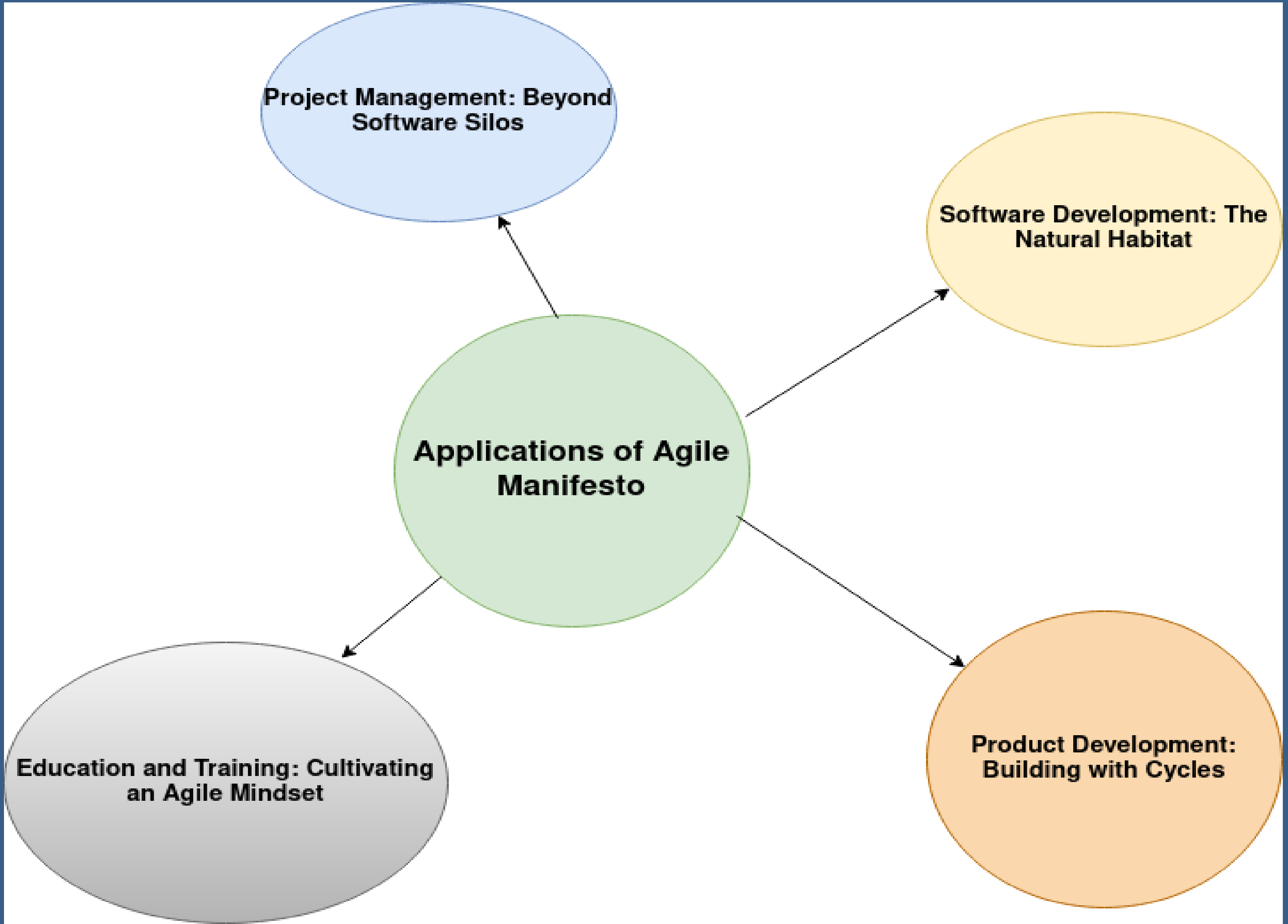
Embrace the Change Wave: Flexibility over Rigidity

Working Software is King: Documentation Takes a Backseat

Teamwork Triumphs:

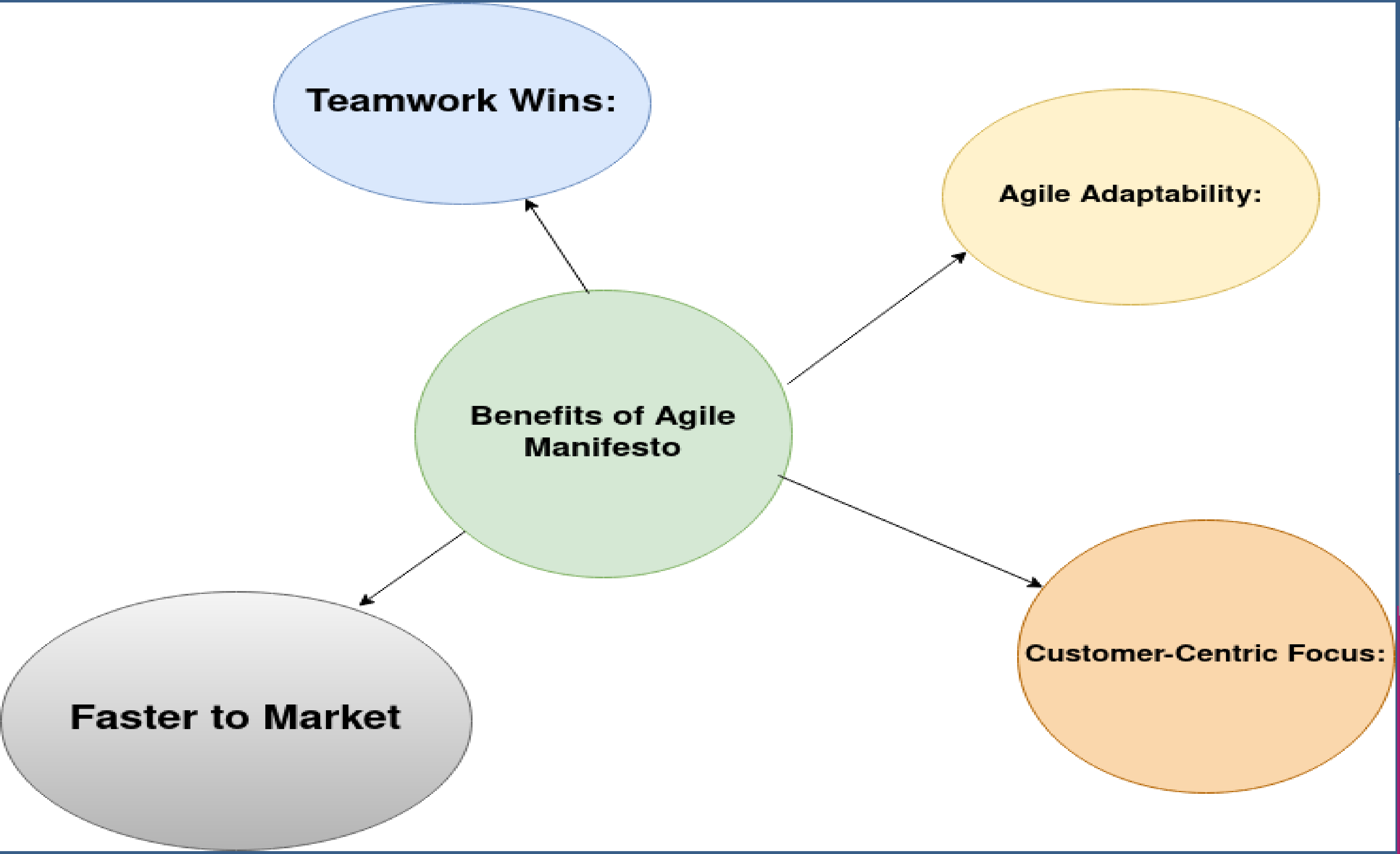
Collaborating with Everyone

APPLICATIONS OF AGILE MANIFESTO FOR SOFTWARE DEVELOPMENT



Project Management: Beyond Software Silos
Software Development: The Natural Habitat
Applications of Agile Manifesto
Education and Training: Cultivating an Agile Mindset
Product Development: Building with Cycles

BENEFITS OF AGILE MANIFESTO



Benefits of Agile Manifesto

Teamwork Wins:

Agile Adaptability:

Customer-Centric Focus:

Faster to Market