CASE STUDY 46:
SkillCraft Online Learning Management System

MALINI 717823L129
ECE-A

1. Low-Level Design (LLD):

1. TABLES WITH ATTRIBUTES AND KEYS:

1. Attributes and Data Types:

1. STUDENTS Table:

ATTRIBUTES	DATA TYPES
student_id	INT (PK)
Name	VARCHAR(50)
Email	VARCHAR(100)
Phone	VARCHAR(10)
join_date	DATE

1.1.1.2. INSTRUCTORS:

ATTRIBUTES	DATA TYPES
instructor_id	INT(PK)
Name	VARCHAR(100)
specialization	VARCHAR(100)
salary	DECIMAL(10,2)

1.1.1.3. COURSES:

ATTRIBUTES	DATA TYPES
course_id	INT(PK)
title	VARCHAR(150)
level	VARCHAR(50)
Category	VARCHAR(100)
instructor_id	INT(FK)

1.1.1.4. MODULES:

ATTRIBUTES	DATA TYPES
module_id	INT(PK)
course_id	INT(PK)
title	VARCHAR(150)
order_no	INT

1.1.1.5. LESSONS:

ATTRIBUTES	DATA TYPES
lesson_id	INT(PK)
module_id	INT(PK)
Title	VARCHAR(150)
duration	INT
type	VARCHAR(50)

1.1.1.6. ENROLLMENTS:

ATTRIBUTES	DATA TYPES
enrollment_id	INT(PK)
student_id	INT(PK)
course_id	INT(PK)
enroll_date	INT
status	VARCHAR(50)

1.1.1.7. PAYMENTS:

ATTRIBUTES	DATA TYPES
payment_id	INT(PK)
enrollment_id	INT(PK)
amount	DECIMAL(10,2)
payment_date	DATE
status	VARCHAR(50)

1.1.1.8. ASSESSMENTS:

ATTRIBUTES	DATA TYPES
assessment_id	INT(PK)
course_id	INT(PK)
type	VARCHAR(50)
total_marks	INT

1.1.1.9. ASSESSMENT_SCORES:

ATTRIBUTES	DATA TYPES
score_id	INT(PK)
assessment_id	INT(FK)
student_id	INT(FK)
marks	INT
attempt_no	INT

1.1.1.10. CERTIFICATES:

ATTRIBUTES	DATA TYPES
cert_id	INT(PK)
student_id	INT(FK)
course_id	INT(FK)
issue_date	DATE

1.1.1.10. CERTIFICATES

ATTRIBUTES	DATA TYPES
cert_id	INT(PK)
student_id	INT(FK)
course_id	INT(FK)
issue_date	DATE

2.Constraints Identification

1.1.2.1. STUDENTS:

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	student_id
Unique	Email
Unique	Phone
Check	join_date IS NOT NULL

1.1.2.2. INSTRUCTORS

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	instructor_id
Unique	email
Check	salary >= 0

1.1.2.3. COURSES

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	course_id
Foreign Key	instructor_id REFERENCES instructors(instructor id)
Check	level IN ('Beginner','Intermediate','Advanced')

1.1.2.4. MODULES

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	module_id
Foreign Key	course_id REFERENCES courses(course_id)
check	order_no >= 1

1.1.2.5. LESSONS

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	lesson_id
Foreign Key	module_id REFERENCES modules(module_id)
Check	duration > 0

1.1.2.6. ENROLLMENTS

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	enrollment_id
Foreign Key	student_id REFERENCES students(student_id)
Foreign Key	course_id REFERENCES courses(course_id)
Check	status IN ('active','completed','dropped')

1.1.2.7. PAYMENTS

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	payment_id
Foreign Key	enrollment_id REFERENCES enrollments(entrollment_id)
Check	status IN ('paid','refunded','failed')
Check	amount >= 0

1.1.2.8. ASSESSMENTS

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	assessment_id
Foreign Key	course_id REFERENCES courses(course_id)
Check	total_marks > 0

1.1.2.9. ASSESSMENT_SCORES

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	score_id
Foreign Key	assessment_id REFERENCES assessments(assessment_id)
Foreign Key	student_id REFERENCES students(student_id)
Check	marks >= 0

1.1.2.10. CERTIFICATES

CONSTRAINT TYPE	ATTRIBUTES
Primary Key	cert_id
Foreign Key	student_id REFERENCES students(student_id)
Foreign Key	course_id REFERENCES courses(course_id)
Check	issue_date IS NOT NULL

2. RELATIONSHIP MAPPING BETWEEN ENTITIES:

1.2.1. ENROLLMENTS → STUDENTS

Child Table	Parent Table	Relationship
ENROLLMENT	STUDENTS	enrollments.student_id → students.student_id

1.2.2. ENROLLMENTS→ COURSES:

Child Table	Parent Table	Relationship
ENROLLMENTS	COURSES	enrollments.course_id → courses.course_id

1.2.3. COURSES → INSTRUCTORS:

Child Table	Parent Table	Relationship
COURSES	INSTRUCTORS	courses.instructor_id → instructors.instructor_id

1.2.4. MODULES → COURSES :

Child Table	Parent Table	Relationship
MODULES	COURSES	modules.course_id → courses.course_id

1.2.5. LESSONS → MODULES:

Child Table	Parent Table	Relationship
LESSONS	MODULES	lessons.module_id → modules.module_id

1.2.6. PAYMENTS→ENROLLMENTS:

Child Table	Parent Table	Relationship
PAYMENTS	ENROLLMENTS	payments.enrollment_id → enrollments.enrollment_id

1.2.7. ASSESSMENTS→COURSES:

Child Table	Parent Table	Relationship
ASSESSMENTS	COURSES	assessments.course_id → courses.course_id

1.2.8. ASSESSMENT_SCORE→ASSESSMENT:

Child Table	Parent Table	Relationship
ASSESSMENT_SCORE	ASSESSMENT	assessment_scores.assessment_id → assessments.assessment_id

1.2.9. ASSESSMENT_SCORE→STUDENT:

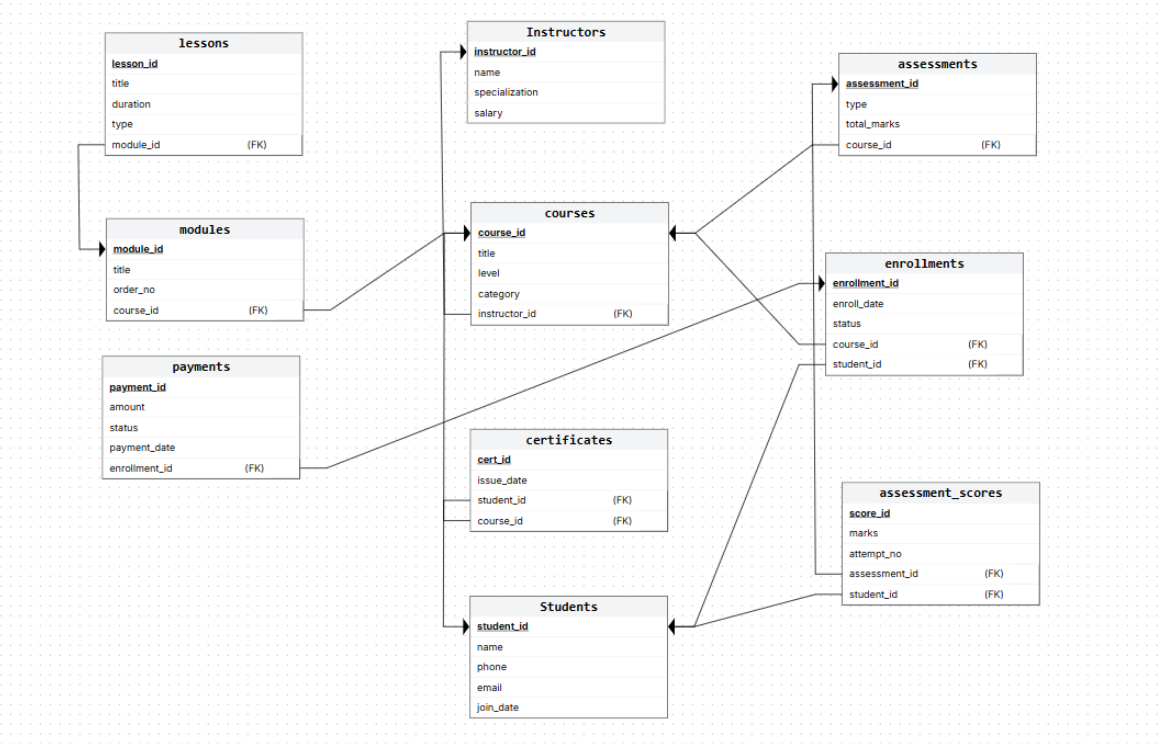
Child Table	Parent Table	Relationship
ASSESSMENT_SCORE	STUDENT	assessment_scores.student_id → students.student_id

1.2.10. CERTIFICATES→STUDENT:

Child Table	Parent Table	Relationship
CERTIFICATES	STUDENT	certificates.student_id → students.student_id

1.2.11. CERTIFICATES→COURSES:

Child Table	Parent Table	Relationship
CERTIFICATES	COURSES	certificates.course_id → courses.course_id



3.**NORMALIZATION OF SKILLCRAFT DATABASE**:

. 1NF (First Normal Form):

• Each table has only one value in each column.

• No repeating groups or multiple values inside a single column.

• Example: In students, one email and one phone number per row.

• Enrollments has one student, one course, and one status per enrollment row.

• Payments records one payment per row with a single status value.

• Assessment_scores stores one student's marks for one assessment attempt per row.

2. 2NF (Second Normal Form):

• Table is already in 1NF.

• All non-key attributes fully depend on the primary key.

• No partial dependency on part of a composite key.

Examples:

• In **enrollments**, attributes like enroll_date and status depend entirely on enrollment_id, not partially on student_id or course_id.

• In **assessment_scores**, marks and attempt_no depend fully on score_id (or on the full composite key if used), not only on student_id or assessment_id.

• In **courses**, title, level, and category depend completely on course_id.

3. 3NF (Third Normal Form):

• Table is already in 2NF.

• No transitive dependency (non-key attributes should not depend on other non-key attributes).

• Every non-key attribute depends only on the primary key.

Examples:

• In **courses**, instructor_name is not stored (to avoid dependency on instructor_id). Instructor details are stored separately in the instructors table.

• In **students**, no course-related data is stored directly; course details are accessed through enrollments.

• In **payments**, payment status depends only on payment_id, not indirectly through enrollment details.

BCNF (Boyce-Codd Normal Form):

• For every functional dependency, the determinant is a candidate key.

• Ensures strong data consistency and removes anomalies.

4. INDEXING STRATEGY NOTES:

Create indexes on student_id, payment_date, and course_id:

```
CREATE INDEX idx_enrollment_student
ON enrollments (student_id);
CREATE INDEX idx_payment_date
ON payments (payment_date);
CREATE INDEX idx_course_id
ON courses (course_id);
```

Rewrite 2 queries (from the list above) into optimized version:

Query 1 – Students enrolled today

```
SELECT enrollment_id, student_id, course_id
FROM enrollments
WHERE enroll_date = CURDATE();
```

Query 2 – Students with instructor and course details (Optimized JOIN)

```
SELECT s.name AS student_name,
    c.title AS course_title,
    i.name AS instructor_name
FROM enrollments e
JOIN students s ON s.student_id = e.student_id
JOIN courses c ON c.course_id = e.course_id
JOIN instructors i ON i.instructor_id = c.instructor_id;
```

SECURITY ROLE DESIGN:

1. ADMIN_ROLE – FULL PRIVILEGES:

```
CREATE ROLE admin_role;

GRANT ALL PRIVILEGES
ON skillcraft_system.*
TO admin_role;
```

2. INSTRUCTOR_ROLE – Manage Courses & Assessments Only:

```
CREATE ROLE instructor_role;

GRANT SELECT, INSERT, UPDATE, DELETE
ON skillcraft_system.courses
TO instructor_role;

GRANT SELECT, INSERT, UPDATE, DELETE
ON skillcraft_system.modules
TO instructor_role;

GRANT SELECT, INSERT, UPDATE, DELETE
ON skillcraft_system.lessons
TO instructor_role;
```

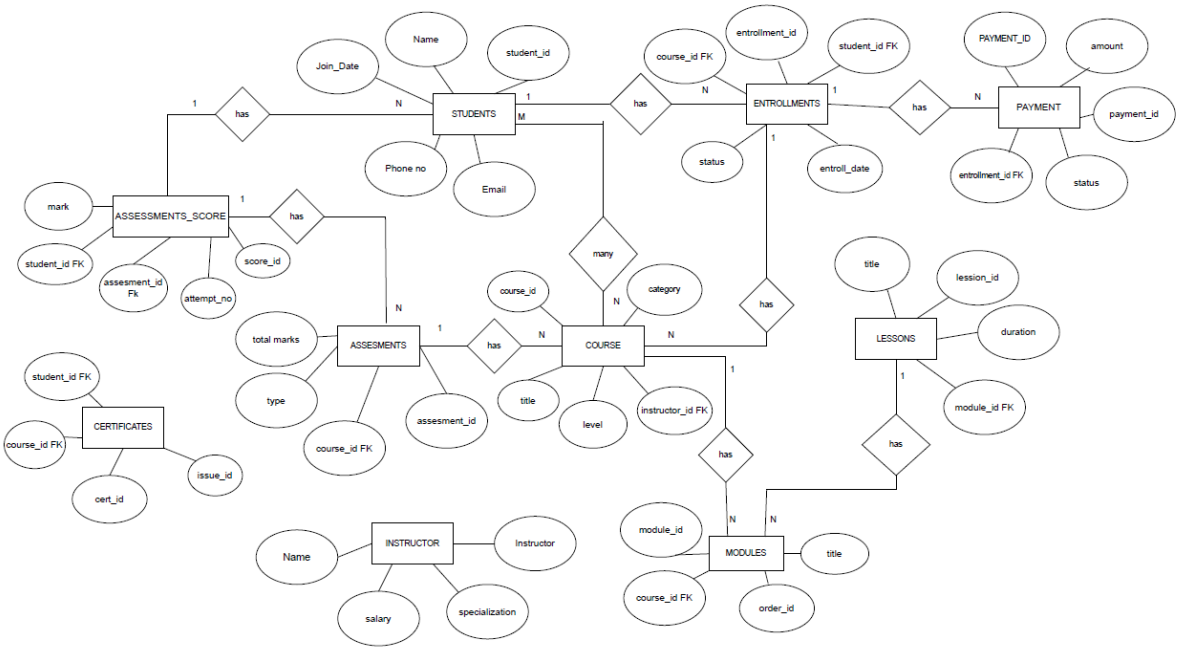
```
GRANT SELECT, INSERT, UPDATE, DELETE
ON skillcraft_system.lessons
TO instructor_role;

GRANT SELECT, INSERT, UPDATE, DELETE
ON skillcraft_system.assessments
TO instructor_role;

GRANT SELECT, INSERT, UPDATE
ON skillcraft_system.assessment_scores
TO instructor_role;
(Instructors cannot access payments table)
```

3. STUDENT_ROLE – Read-Only Access to Their Progress:

```
CREATE ROLE student_role;

GRANT SELECT
ON skillcraft_system.enrollments
TO student_role;

GRANT SELECT
ON skillcraft_system.assessment_scores
TO student_role;

GRANT SELECT
ON skillcraft_system.certificates
TO student_role;
```

MAIN REPORT

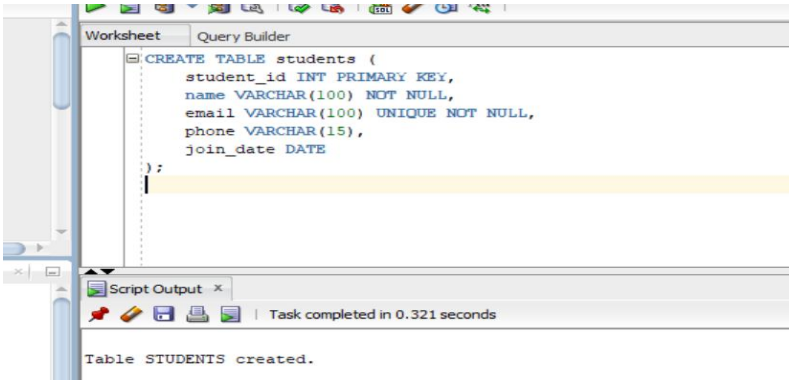
2.1 ER diagram:



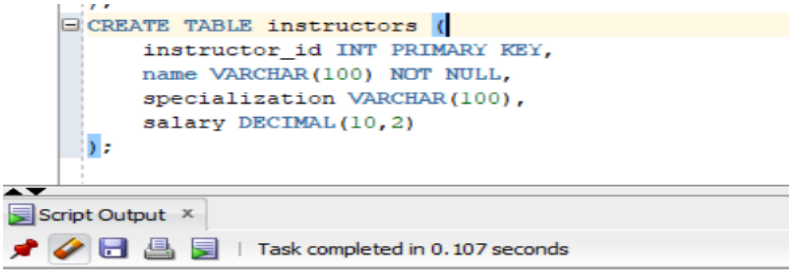
2.2 SQL code with screenshots && Explanation :

Create table:

Students:



Instructors:




```
Table STUDENTS created.


Table INSTRUCTORS created.
```

Courses:

```
);
CREATE TABLE courses (
    course_id NUMBER PRIMARY KEY,
    title VARCHAR2(150) NOT NULL,
    course_level VARCHAR2(50),
    category VARCHAR2(100),
    instructor_id NUMBER,
    FOREIGN KEY (instructor_id) REFERENCES instructors(instructor_id)
);
```

Script Output | Task completed in 0.095 seconds

```
    FOREIGN KEY (instructor_id) REFERENCES instructors(instructor_id)
)
Error report -
ORA-00904: : invalid identifier

https://docs.oracle.com/error-help/db/ora-00904/00904. 00000 -  "%s: invalid i
*Cause:    The identifier or column name entered was invalid.
*Action:   Ensure the following

Table COURSES created.
```

Modules:

```
);
CREATE TABLE modules (
    module_id INT PRIMARY KEY,
    title VARCHAR(150) NOT NULL,
    order_no INT,
    course_id INT,
    FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
```

Script Output | Task completed in 0.055 seconds

```
ORA-00904: : invalid identifier

https://docs.oracle.com/error-help/db/ora-00904/00904. 00000 -  "%s: invali
*Cause:    The identifier or column name entered was invalid.
*Action:   Ensure the following

Table COURSES created.


Table MODULES created.
```

Lessons:

```
);
CREATE TABLE lessons (
    lesson_id INT PRIMARY KEY,
    title VARCHAR(150) NOT NULL,
    duration INT,
    type VARCHAR(50),
    module_id INT,
    FOREIGN KEY (module_id) REFERENCES modules(module_id)
);
```

Script Output | Task completed in 0.115 seconds

```
*Cause:    The identifier or column name entered was invalid.
*Action:   Ensure the following

Table COURSES created.


Table MODULES created.


Table LESSONS created.
```

Enrollments:

```
CREATE TABLE enrollments (
    enrollment_id INT PRIMARY KEY,
    enroll_date DATE,
    status VARCHAR(50),
    course_id INT,
    student_id INT,
    FOREIGN KEY (course_id) REFERENCES courses(course_id),
    FOREIGN KEY (student_id) REFERENCES students(student_id)
);
```

Script Output | Task completed in 0.06 seconds

```
Table ENROLLMENTS created.
```

Payments:

```
CREATE TABLE payments (
    payment_id INT PRIMARY KEY,
    amount DECIMAL(10,2),
    status VARCHAR(20) CHECK (status IN ('paid','refunded','failed')),
    payment_date DATE,
    enrollment_id INT,
    FOREIGN KEY (enrollment_id) REFERENCES enrollments(enrollment_id)
);
```

Script Output | Task completed in 0.145 seconds

```
Table PAYMENTS created.
```

Assessments:

```
CREATE TABLE assessments (
    assessment_id INT PRIMARY KEY,
    type VARCHAR(50),
    total_marks INT,
    course_id INT,
    FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
```

Script Output | Task completed in 0.054 seconds

```
Table ASSESSMENTS created.
```

Assessments_score:

```
CREATE TABLE assessment_scores (
    score_id INT PRIMARY KEY,
    marks INT,
    attempt_no INT,
    assessment_id INT,
    student_id INT,
    FOREIGN KEY (assessment_id) REFERENCES assessments(assessment_id),
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    CHECK (marks >= 0)
);
```

Script Output | Task completed in 0.103 seconds

```
Table ASSESSMENT_SCORES created.
```

Certificates:

```
CREATE TABLE certificates (
    cert_id INT PRIMARY KEY,
    issue_date DATE,
    student_id INT,
    course_id INT,
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
```

Script Output | Task completed in 0.075 seconds

```
Table CERTIFICATES created.
```

Insert table:

Students:

```
INSERT INTO students VALUES (5,'Arjun Rao','arjun5@gmail.com','9000000005',DATE '2024-01-20');
INSERT INTO students VALUES (6,'Sneha Iyer','sneha6@gmail.com','9000000006',DATE '2024-02-01');
INSERT INTO students VALUES (7,'Kiran Patel','kiran7@gmail.com','9000000007',DATE '2024-02-05');
INSERT INTO students VALUES (8,'Pooja Nair','pooja8@gmail.com','9000000008',DATE '2024-02-10');
INSERT INTO students VALUES (9,'Rohit Das','rohit9@gmail.com','9000000009',DATE '2024-02-15');
INSERT INTO students VALUES (10,'Meera Joshi','meera10@gmail.com','9000000010',DATE '2024-02-20');
INSERT INTO students VALUES (11,'Aman Gupta','aman11@gmail.com','9000000011',DATE '2024-03-01');
INSERT INTO students VALUES (12,'Divya Roy','divya12@gmail.com','9000000012',DATE '2024-03-05');
INSERT INTO students VALUES (13,'Sanjay Kumar','sanjay13@gmail.com','9000000013',DATE '2024-03-10');
```

All Rows Fetched: 20 in 0.081 seconds

	STUDENT_ID	NAME	EMAIL	PHONE	JOIN_DATE
1	1	Rahul Kumar	rahul1@gmail.com	9000000001	01-01-24
2	2	Anita Sharma	anita2@gmail.com	9000000002	05-01-24
3	3	Vikram Singh	vikram3@gmail.com	9000000003	10-01-24
4	4	Neha Verma	neha4@gmail.com	9000000004	15-01-24
5	5	Arjun Rao	arjun5@gmail.com	9000000005	20-01-24
6	6	Sneha Iyer	sneha6@gmail.com	9000000006	01-02-24
7	7	Kiran Patel	kiran7@gmail.com	9000000007	05-02-24
8	8	Pooja Nair	pooja8@gmail.com	9000000008	10-02-24
9	9	Rohit Das	rohit9@gmail.com	9000000009	15-02-24
10	10	Meera Joshi	meera10@gmail.com	9000000010	20-02-24
11	11	Aman Gupta	aman11@gmail.com	9000000011	01-03-24

Instructors

```
INSERT INTO instructors VALUES (111,'Bhavna Rao','ReactJS',62000);
INSERT INTO instructors VALUES (112,'Kunal Shah','NodeJS',61000);
INSERT INTO instructors VALUES (113,'Ritu Das','SQL',57000);
INSERT INTO instructors VALUES (114,'Siddharth Roy','Testing',54000);
INSERT INTO instructors VALUES (115,'Varsha Iyer','Networking',63000);
INSERT INTO instructors VALUES (116,'Arvind Patel','Blockchain',82000);
INSERT INTO instructors VALUES (117,'Neeraj Yadav','Big Data',76000);
INSERT INTO instructors VALUES (118,'Pallavi Sen','UI Research',59000);
INSERT INTO instructors VALUES (119,'Harsha Kumar','Android Dev',68000);
INSERT INTO instructors VALUES (120,'Komal Verma','iOS Dev',69000);
SELECT * FROM instructors;
```

All Rows Fetched: 20 in 0.02 seconds

	INSTRUCTOR_ID	NAME	SPECIALIZATION	SALARY
1	101	Suresh Reddy	Web Development	55000
2	102	Priya Menon	UI/UX	60000
3	103	Ramesh Gupta	Data Science	70000
4	104	Anjali Kapoor	Cyber Security	65000
5	105	Vikas Sharma	Cloud Computing	72000
6	106	Manoj Kumar	Python	58000
7	107	Swati Jain	Java	60000
8	108	Tarun Mehta	AI	80000
9	109	Rekha Nair	Machine Learning	75000
10	110	Ajay Singh	DevOps	73000
11	111	Bhavna Rao	ReactJS	62000

Courses

```
INSERT INTO courses VALUES (209,'Machine Learning Pro','Advanced','Certification',109);
INSERT INTO courses VALUES (210,'DevOps Complete','Advanced','Certification',110);
INSERT INTO courses VALUES (211,'ReactJS','Intermediate','Programming',111);
INSERT INTO courses VALUES (212,'NodeJS','Intermediate','Programming',112);
INSERT INTO courses VALUES (213,'SQL Basics','Beginner','Database',113);
INSERT INTO courses VALUES (214,'Software Testing','Beginner','Testing',114);
INSERT INTO courses VALUES (215,'Networking','Intermediate','Infrastructure',115);
INSERT INTO courses VALUES (216,'Blockchain Intro','Advanced','Certification',116);
INSERT INTO courses VALUES (217,'Big Data Hadoop','Advanced','Certification',117);
INSERT INTO courses VALUES (218,'UX Research','Intermediate','Design',118);
INSERT INTO courses VALUES (219,'Android Dev','Intermediate','Mobile',119);
INSERT INTO courses VALUES (220,'iOS Development','Intermediate','Mobile',120);
SELECT * FROM courses;
```

All Rows Fetched: 20 in 0.007 seconds

	COURSE_ID	TITLE	COURSE_LEVEL	CATEGORY	INSTRUCTOR_ID
1	201	Full Stack Web	Advanced	Certification	101
2	202	UI Design Basics	Beginner	Design	102
3	203	Python Programming	Intermediate	Programming	106
4	204	Data Science Bootcamp	Advanced	Certification	103
5	205	Cyber Security Fundamentals	Intermediate	Security	104
6	206	Cloud AWS	Advanced	Certification	105
7	207	Java Masterclass	Intermediate	Programming	107
8	208	AI Basics	Beginner	AI	108
9	209	Machine Learning Pro	Advanced	Certification	109
10	210	DevOps Complete	Advanced	Certification	110
11	211	ReactJS	Intermediate	Programming	111
12	212	NodeJS	Intermediate	Programming	112

Modules:

```
INSERT INTO modules VALUES (307,'AWS Setup',1,206);
INSERT INTO modules VALUES (308,'Java OOP',1,207);
INSERT INTO modules VALUES (309,'AI Intro',1,208);
INSERT INTO modules VALUES (310,'ML Algorithms',1,209);
INSERT INTO modules VALUES (311,'CI/CD',1,210);
INSERT INTO modules VALUES (312,'React Components',1,211);
INSERT INTO modules VALUES (313,'ExpressJS',1,212);
INSERT INTO modules VALUES (314,'SQL Queries',1,213);
INSERT INTO modules VALUES (315,'Manual Testing',1,214);
INSERT INTO modules VALUES (316,'Network Basics',1,215);
INSERT INTO modules VALUES (317,'Blockchain Core',1,216);
INSERT INTO modules VALUES (318,'Hadoop Intro',1,217);
INSERT INTO modules VALUES (319,'User Research',1,218);
INSERT INTO modules VALUES (320,'Android UI',1,219);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.003 seconds

	MODULE_ID	TITLE	ORDER_NO	COURSE_ID
1	301	HTML Basics	1	201
2	302	CSS Styling	2	201
3	303	Figma Intro	1	202
4	304	Advanced Python	1	203
5	305	ML Concepts	1	204
6	306	Security Threats	1	205
7	307	AWS Setup	1	206
8	308	Java OOP	1	207
9	309	AI Intro	1	208
10	310	ML Algorithms	1	209
11	311	CI/CD	1	210

Lessons:

```
INSERT INTO lessons VALUES (408,'Java Classes',50,'Recorded',308);
INSERT INTO lessons VALUES (409,'AI Applications',60,'Recorded',309);
INSERT INTO lessons VALUES (410,'Decision Trees',65,'Live',310);
INSERT INTO lessons VALUES (411,'Pipeline Setup',75,'Live',311);
INSERT INTO lessons VALUES (412,'Hooks in React',60,'Recorded',312);
INSERT INTO lessons VALUES (413,'Routing',45,'Recorded',313);
INSERT INTO lessons VALUES (414,'Select Statement',50,'Live',314);
INSERT INTO lessons VALUES (415,'Test Cases',40,'Recorded',315);
INSERT INTO lessons VALUES (416,'OSI Model',55,'Recorded',316);
INSERT INTO lessons VALUES (417,'Smart Contracts',65,'Live',317);
INSERT INTO lessons VALUES (418,'MapReduce',70,'Recorded',318);
INSERT INTO lessons VALUES (419,'User Interviews',50,'Live',319);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.026 seconds

	LESSON_ID	TITLE	DURATION	TYPE	MODULE_ID
1	401	HTML Tags	60	Recorded	301
2	402	CSS Flexbox	45	Live	302
3	403	Wireframing	50	Recorded	303
4	404	Loops in Python	55	Live	304
5	405	Regression	60	Recorded	305
6	406	Malware Types	40	Recorded	306
7	407	EC2 Setup	70	Live	307
8	408	Java Classes	50	Recorded	308
9	409	AI Applications	60	Recorded	309
10	410	Decision Trees	65	Live	310

enrollments:

```
INSERT INTO enrollments VALUES (508,DATE '2024-02-08','Active',208,8);
INSERT INTO enrollments VALUES (509,DATE '2024-02-09','Completed',209,9);
INSERT INTO enrollments VALUES (510,DATE '2024-02-10','Active',210,10);
INSERT INTO enrollments VALUES (511,DATE '2024-02-11','Active',211,11);
INSERT INTO enrollments VALUES (512,DATE '2024-02-12','Completed',212,12);
INSERT INTO enrollments VALUES (513,DATE '2024-02-13','Active',213,13);
INSERT INTO enrollments VALUES (514,DATE '2024-02-14','Active',214,14);
INSERT INTO enrollments VALUES (515,DATE '2024-02-15','Completed',215,15);
INSERT INTO enrollments VALUES (516,DATE '2024-02-16','Active',216,16);
INSERT INTO enrollments VALUES (517,DATE '2024-02-17','Active',217,17);
INSERT INTO enrollments VALUES (518,DATE '2024-02-18','Completed',218,18);
INSERT INTO enrollments VALUES (519,DATE '2024-02-19','Active',219,19);
INSERT INTO enrollments VALUES (520,DATE '2024-02-20','Active',220,20);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.014 seconds

	ENROLLMENT_ID	ENROLL_DATE	STATUS	COURSE_ID	STUDENT_ID
1	501	01-02-24	Active	201	1
2	502	02-02-24	Active	202	2
3	503	03-02-24	Completed	203	3
4	504	04-02-24	Active	204	4
5	505	05-02-24	Active	205	5
6	506	06-02-24	Completed	206	6
7	507	07-02-24	Active	207	7
8	508	08-02-24	Active	208	8
9	509	09-02-24	Completed	209	9
10	510	10-02-24	Active	210	10
11	511	11-02-24	Active	211	11
12	512	12-02-24	Completed	212	12

payments:

```
INSERT INTO payments VALUES (607,13000,'paid',DATE '2024-02-07',507);
INSERT INTO payments VALUES (608,11000,'paid',DATE '2024-02-08',508);
INSERT INTO payments VALUES (609,17000,'paid',DATE '2024-02-09',509);
INSERT INTO payments VALUES (610,16000,'paid',DATE '2024-02-10',510);
INSERT INTO payments VALUES (611,12500,'paid',DATE '2024-02-11',511);
INSERT INTO payments VALUES (612,11500,'paid',DATE '2024-02-12',512);
INSERT INTO payments VALUES (613,9000,'paid',DATE '2024-02-13',513);
INSERT INTO payments VALUES (614,9500,'paid',DATE '2024-02-14',514);
INSERT INTO payments VALUES (615,10500,'paid',DATE '2024-02-15',515);
INSERT INTO payments VALUES (616,22000,'paid',DATE '2024-02-16',516);
INSERT INTO payments VALUES (617,21000,'paid',DATE '2024-02-17',517);
INSERT INTO payments VALUES (618,10000,'paid',DATE '2024-02-18',518);
INSERT INTO payments VALUES (619,14000,'paid',DATE '2024-02-19',519);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.006 seconds

	PAYMENT_ID	AMOUNT	STATUS	PAYMENT_DATE	ENROLLMENT_ID
1	601	15000	paid	01-02-24	501
2	602	10000	paid	02-02-24	502
3	603	12000	paid	03-02-24	503
4	604	18000	paid	04-02-24	504
5	605	14000	paid	05-02-24	505
6	606	20000	paid	06-02-24	506
7	607	13000	paid	07-02-24	507
8	608	11000	paid	08-02-24	508
9	609	17000	paid	09-02-24	509
10	610	16000	paid	10-02-24	510
11	611	12500	paid	11-02-24	511

Assessment:

```
INSERT INTO assessments VALUES (709,'Quiz',100,209);
INSERT INTO assessments VALUES (710,'Assignment',100,210);
INSERT INTO assessments VALUES (711,'Quiz',100,211);
INSERT INTO assessments VALUES (712,'Assignment',100,212);
INSERT INTO assessments VALUES (713,'Quiz',100,213);
INSERT INTO assessments VALUES (714,'Assignment',100,214);
INSERT INTO assessments VALUES (715,'Quiz',100,215);
INSERT INTO assessments VALUES (716,'Assignment',100,216);
INSERT INTO assessments VALUES (717,'Quiz',100,217);
INSERT INTO assessments VALUES (718,'Assignment',100,218);
INSERT INTO assessments VALUES (719,'Quiz',100,219);
INSERT INTO assessments VALUES (720,'Assignment',100,220);
SELECT * FROM assessments
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.005 seconds

	ASSESSMENT_ID	TYPE	TOTAL_MARKS	COUR...
1	701	Quiz	100	201
2	702	Assignment	100	202
3	703	Quiz	100	203
4	704	Quiz	100	204
5	705	Assignment	100	205
6	706	Quiz	100	206
7	707	Quiz	100	207
8	708	Assignment	100	208
9	709	Quiz	100	209
10	710	Assignment	100	210
11	711	Quiz	100	211

Assessment_scores:

```
INSERT INTO assessment_scores VALUES (811,84,1,711,11);
INSERT INTO assessment_scores VALUES (812,73,1,712,12);
INSERT INTO assessment_scores VALUES (813,69,1,713,13);
INSERT INTO assessment_scores VALUES (814,91,1,714,14);
INSERT INTO assessment_scores VALUES (815,77,1,715,15);
INSERT INTO assessment_scores VALUES (816,93,1,716,16);
INSERT INTO assessment_scores VALUES (817,88,1,717,17);
INSERT INTO assessment_scores VALUES (818,74,1,718,18);
INSERT INTO assessment_scores VALUES (819,81,1,719,19);
INSERT INTO assessment_scores VALUES (820,86,1,720,20);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.015 seconds

	SCORE_ID	MARKS	ATTEMPT_NO	ASSESSMENT_ID	STUDENT_ID
1	801	85	1	701	1
2	802	78	1	702	2
3	803	90	1	703	3
4	804	88	1	704	4
5	805	67	1	705	5
6	806	92	1	706	6
7	807	75	1	707	7
8	808	80	1	708	8
9	809	89	1	709	9
10	810	76	1	710	10

Certificates:

```
INSERT INTO certificates VALUES (907,DATE '2024-05-07',1,201);
INSERT INTO certificates VALUES (908,DATE '2024-05-08',2,202);
INSERT INTO certificates VALUES (909,DATE '2024-05-09',4,204);
INSERT INTO certificates VALUES (910,DATE '2024-05-10',5,205);
INSERT INTO certificates VALUES (911,DATE '2024-05-11',7,207);
INSERT INTO certificates VALUES (912,DATE '2024-05-12',8,208);
INSERT INTO certificates VALUES (913,DATE '2024-05-13',10,210);
INSERT INTO certificates VALUES (914,DATE '2024-05-14',11,211);
INSERT INTO certificates VALUES (915,DATE '2024-05-15',13,213);
INSERT INTO certificates VALUES (916,DATE '2024-05-16',14,214);
INSERT INTO certificates VALUES (917,DATE '2024-05-17',16,216);
INSERT INTO certificates VALUES (918,DATE '2024-05-18',17,217);
INSERT INTO certificates VALUES (919,DATE '2024-05-19',19,219);
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.012 seconds

	CERT_ID	ISSUE_DATE	STUDENT_ID	COURSE_ID
1	901	01-05-24	3	203
2	902	02-05-24	6	206
3	903	03-05-24	9	209
4	904	04-05-24	12	212
5	905	05-05-24	15	215
6	906	06-05-24	18	218
7	907	07-05-24	1	201
8	908	08-05-24	2	202
9	909	09-05-24	4	204
10	910	10-05-24	5	205
11	911	11-05-24	7	207

BASICS QUERIES:

1.List students who joined in 2024.

```
SELECT *
FROM students
WHERE join_date BETWEEN DATE '2024-01-01' AND DATE '2024-12-31';
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0.06 seconds

	STUDENT_ID	NAME	EMAIL	PHONE	JOIN_DATE
1	1	Rahul Kumar	rahul1@gmail.com	9000000001	01-01-24
2	2	Anita Sharma	anita2@gmail.com	9000000002	05-01-24
3	3	Vikram Singh	vikram3@gmail.com	9000000003	10-01-24
4	4	Neha Verma	neha4@gmail.com	9000000004	15-01-24
5	5	Arjun Rao	arjun5@gmail.com	9000000005	20-01-24
6	6	Sneha Iyer	sneha6@gmail.com	9000000006	01-02-24
7	7	Kiran Patel	kiran7@gmail.com	9000000007	05-02-24
8	8	Pooja Nair	pooja8@gmail.com	9000000008	10-02-24
9	9	Rohit Das	rohit9@gmail.com	9000000009	15-02-24
10	10	Meera Joshi	meera10@gmail.com	9000000010	20-02-24
11	11	Aman Gupta	aman11@gmail.com	9000000011	01-03-24
12	12	Divya Roy	divya12@gmail.com	9000000012	05-03-24

Explanation: selects all students whose join date is between Jan 1 and Dec 31 of 2024

2.Instructors without any assigned course

```
SELECT *
FROM instructors
WHERE instructor_id NOT IN
      (SELECT instructor_id FROM courses);
```

Script Output | Query Result

SQL | All Rows Fetched: 0 in 0.038 seconds

INSTRUC...	NAME	SPECI...	SALARY

Filter...

Explanation:Instructors not teaching any course

3.Courses categorized as “Advanced”.

```
SELECT *
FROM courses
WHERE course_level = 'Advanced';
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.01 seconds

	COURSE_ID	TITLE	COURSE_LEVEL	CATEGORY	INSTRUCTOR_ID
1	201	Full Stack Web	Advanced	Certification	101
2	204	Data Science Bootcamp	Advanced	Certification	103
3	206	Cloud AWS	Advanced	Certification	105
4	209	Machine Learning Pro	Advanced	Certification	109
5	210	DevOps Complete	Advanced	Certification	110
6	216	Blockchain Intro	Advanced	Certification	116
7	217	Big Data Hadoop	Advanced	Certification	117

Explanation : Courses with Advanced level

4.Students who failed an assessment (marks < 40%).

```
ALTER TABLE assessment_scores
ADD result VARCHAR2(10);
UPDATE assessment_scores
SET marks = 30
WHERE score_id IN (805, 812, 818);


SELECT student_id
FROM assessment_scores
WHERE marks < 40;
```

Script Output | Query Result

SQL | All Rows Fetched: 3 in 0.004 seconds

	STUDENT_ID
1	5
2	12
3	18

Explanation : Students scoring below 40 marks

5.Students enrolled in more than 3 courses.

```
INSERT INTO enrollments VALUES (521, DATE '2024-03-01', 'Active', 202, 1);

INSERT INTO enrollments VALUES (522, DATE '2024-03-05', 'Active', 203, 1);

INSERT INTO enrollments VALUES (523, DATE '2024-03-10', 'Active', 204, 1);
SELECT student_id
FROM enrollments
GROUP BY student_id
HAVING COUNT(course_id) > 3;
```

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.007 seconds

	STUDENT_ID
1	1

Explanation : shows students who are enrolled in more than three courses

Joins & Subqueries :

6.Show students with their instructor and course details.

```
SELECT s.name, c.title, i.name
FROM students s
JOIN enrollments e ON s.student_id = e.student_id
JOIN courses c ON e.course_id = c.course_id
JOIN instructors i ON c.instructor_id = i.instructor_id;
```

Script Output | Query Result

SQL | All Rows Fetched: 23 in 0.047 seconds

	NAME	TITLE	NAME_1
4	Neha Verma	Data Science Bootcamp	Ramesh Gupta
5	Arjun Rao	Cyber Security Fundamentals	Anjali Kapoor
6	Sneha Iyer	Cloud AWS	Vikas Sharma
7	Kiran Patel	Java Masterclass	Swati Jain
8	Pooja Nair	AI Basics	Tarun Mehta
9	Rohit Das	Machine Learning Pro	Rekha Nair
10	Meera Joshi	DevOps Complete	Ajay Singh
11	Aman Gupta	ReactJS	Bhavna Rao
12	Divya Roy	NodeJS	Kunal Shah
13	Sanjay Kumar	SQL Basics	Ritu Das
14	Lakshmi Devi	Software Testing	Siddharth Roy
15	Harish Babu	Networking	Varsha Iyer

Explanation:Displays student, course, instructor information

7.Students scoring below course average in a quiz.

```
SELECT student_id, marks
FROM assessment_scores
WHERE marks < (SELECT AVG(marks) FROM assessment_scores);
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.048 seconds

	STUDENT_ID	MARKS
1	5	30
2	7	75
3	12	30
4	13	69
5	18	30

Explanation:Shows students scoring below average marks

8.Instructors teaching both “Web Development” and “UI/UX”.

```
SELECT instructor_id
FROM courses
WHERE title IN ('Full Stack Web','UI Design Basics')
GROUP BY instructor_id
HAVING COUNT(*) = 2;
```

Script Output | Query Result | SQL | All Rows Fetched: 1 in 0.003 seconds

	INSTRUCTOR_ID
1	101

Explanation:Instructors handling both specified courses

9.Courses with > 80% dropout rate.

```
SELECT * FROM enrollments WHERE course_id = 201;
UPDATE enrollments
SET status = 'Dropped'
WHERE course_id = 201
AND enrollment_id IN (501,521,522,523);
```

Script Output | Query Result | Query Result 1 | SQL | All Rows Fetched: 1 in 0.001 seconds

	COURSE_ID
1	201

Explanation:Courses where most students dropped

10.Students who never had a failed payment.

```
SELECT student_id
FROM enrollments
WHERE enrollment_id NOT IN
        (SELECT enrollment_id
         FROM payments
         WHERE status = 'failed');
```

Script Output | Query Result | Query Result 1 | SQL | All Rows Fetched: 23 in 0.018 seconds

	STUDENT_ID
1	1
2	1
3	1
4	4
5	5
6	6
7	7
8	8
9	9

Explanation:Students without any failed payments

Reports :

11.Top 10 students by overall performance.

```
SELECT student_id, AVG(marks) AS average_marks
FROM assessment_scores
GROUP BY student_id;
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 20 in 0.013 seconds

	STUDENT_ID	AVERAGE_MARKS
1	1	85
2	6	92
3	11	84
4	13	69
5	2	78
6	14	91
7	20	86

Explanation:returns the hour with the most bookings.

12.Total revenue per category

```
SELECT c.category, SUM(p.amount) AS total_revenue
FROM payments p
JOIN enrollments e ON p.enrollment_id = e.enrollment_id
JOIN courses c ON e.course_id = c.course_id
WHERE p.status = 'paid'
GROUP BY c.category;
```

visitormanagement : SELECT * FROM enrollments WHERE course_i

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 9 in 0.04 seconds

	CATEGORY	TOTAL_REVENUE
1	Mobile	29000
2	Programming	49000
3	Infrastructure	10500
4	Certification	129000
5	Design	20000
6	Security	14000
7	AI	11000
8	Database	9000
9	Testing	9500

Explanation:Instructor handling the highest number of courses.

13. Instructor teaching maximum number courses.

```
SELECT *
FROM (
    SELECT instructor_id, COUNT(course_id) AS total_courses
    FROM courses
    GROUP BY instructor_id
    ORDER BY total_courses DESC
)
WHERE ROWNUM = 1;
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 1 in 0.046 seconds

	INSTRUCTOR_ID	TOTAL_COURSES
1	101	2

Explanation: Instructor teaching maximum number courses

14.Monthly student activity levels.

```
SELECT TO_CHAR(enroll_date,'MM-YYYY') AS month,
       COUNT(*) AS total_enrollments
FROM enrollments
GROUP BY TO_CHAR(enroll_date,'MM-YYYY');
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 2 in 0.005 seconds

	MONTH	TOTAL_ENROLLMENTS
1	02-2024	20
2	03-2024	3

Explanation: Shows enrollments count for each month

15.Courses with < 50% average assessment score

```
SELECT a.course_id, AVG(s.marks) AS avg_marks
FROM assessment_scores s
JOIN assessments a ON s.assessment_id = a.assessment_id
GROUP BY a.course_id
HAVING AVG(s.marks) < 50;
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 3 in 0.022 seconds

	COURSE_ID	AVG_MARKS
1	218	30
2	205	30
3	212	30

Explanation:Courses with low average marks

Advanced :

16.Monthly salary report (Instructor count + Total payout).

```
SELECT COUNT(*) AS total_instructors,
       SUM(salary) AS total_salary
FROM instructors;
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 1 in 0.044 seconds

	TOTAL_INSTRUCTORS	TOTAL_SALARY
1	20	1319000

Explanation:Counts instructors and total salary

17.Instructors earning above their department average.

.

```
FROM instructors;
SELECT * FROM instructors
WHERE salary > (SELECT AVG(salary) FROM instructors);
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 9 in 0.03 seconds

	INSTRUCTOR_ID	NAME	SPECIALIZATION	SALARY
1	103	Ramesh Gupta	Data Science	70000
2	105	Vikas Sharma	Cloud Computing	72000
3	108	Tarun Mehta	AI	80000
4	109	Rekha Nair	Machine Learning	75000
5	110	Ajay Singh	DevOps	73000
6	116	Arvind Patel	Blockchain	82000
7	117	Neeraj Yadav	Big Data	76000
8	119	Harsha Kumar	Android Dev	68000
9	120	Komal Verma	iOS Dev	69000

Explanation:Shows instructors earning above average

18.Students whose average scores > global average.

```
SELECT student_id, AVG(marks) AS avg_marks
FROM assessment_scores
GROUP BY student_id
HAVING AVG(marks) >
       (SELECT AVG(marks) FROM assessment_scores);
```

Script Output | Query Result | Query Result 1 | Query Result

SQL | All Rows Fetched: 15 in 0.027 seconds

	STUDENT_ID	AVG_MARKS
1	1	85
2	6	92
3	11	84
4	2	78
5	14	91
6	20	86
7	4	88
8	8	80
9	17	88
10	3	90

Explanation:Students scoring higher than overall average

19.Students enrolled only in “Certification Courses”.

```
SELECT student_id
FROM enrollments e
JOIN courses c ON e.course_id = c.course_id
GROUP BY student_id
HAVING COUNT(DISTINCT c.category) = 1
   AND MAX(c.category) = 'Certification';
```

Script Output | Query Result | Query Result 1 | Query Result 2

SQL | All Rows Fetched: 6 in 0.039 seconds

	STUDENT_ID
1	6
2	4
3	17
4	9
5	10
6	16

Explanation:Students enrolled in certification category

20.A view with: Student Name, Course Count, Completion %, Payment Status

```
SELECT s.name,
       COUNT(e.course_id) AS total_courses,
       p.status
FROM students s
JOIN enrollments e ON s.student_id = e.student_id
JOIN payments p ON e.enrollment_id = p.enrollment_id
GROUP BY s.name, p.status;
```

Script Output | Query Result | Query Result 1 | Query Result 2

SQL | All Rows Fetched: 18 in 0.018 seconds

	NAME	TOTAL_COURSES	STATUS
4	Pooja Nair	1	paid
5	Rohit Das	1	paid
6	Nisha Reddy	1	paid
7	Kiran Patel	1	paid
8	Divya Roy	1	paid
9	Lakshmi Devi	1	paid
10	Deepak Yadav	1	paid
11	Arjun Rao	1	paid
12	Sanjay Kumar	1	paid
13	Priyanka Sen	1	paid

Explanation:View showing courses and payment status

2.3 Execution plans (index vs no index) :

Difference:

Feature	Without	With
Access type	All	Ref
Key	Null	idx_student_id
Rows scanned	High	Very low
I /O operations	High	Low
Execution Plan keywords	TABLE SPAN	INDEX SEEK
Scalability	Poor	Excellent
Query speed	Slow	Fast

2.4 Security role enforcement :

Role-Based Access Control (RBAC):
Create Roles:

```
CREATE ROLE admin_role;
CREATE ROLE instructor_role;
CREATE ROLE student_role:
```

Admin → Full Privileges:

```
GRANT ALL PRIVILEGES ON skillcraft_lms.* TO admin_role;
```

Instructor → Manage course content only

```
GRANT SELECT, INSERT, UPDATE
ON skillcraft_lms.courses
TO instructor_role;

GRANT SELECT, INSERT, UPDATE
ON skillcraft_lms.modules
TO instructor_role;

GRANT SELECT, INSERT, UPDATE
ON skillcraft_lms.lessons
TO instructor_role;

GRANT SELECT, INSERT, UPDATE
ON skillcraft_lms.assessments
TO instructor_role;
```

Student → Read-only access to own progress

```
GRANT SELECT
ON skillcraft_lms.enrollments
TO student_role;

GRANT SELECT
ON skillcraft_lms.assessment_scores
TO student_role;

GRANT SELECT
ON skillcraft_lms.certificates
TO student_role;
```