Write DML Statements to simulate the following Business Processes:

1. Setup a department, course within a department with 2 sections. Section must have a location assigned. Produce a report showing the department, its course and sections (with complete section information).

2. Now register a student to a section and process student payment. Produce a report showing student registration information, including payment information.

How can you improve your DB Schema further. Make four recommendations.

```
CREATE TABLE COLLEGE (
id Number(5) NOT NULL,
Name VARCHAR2(20) NOT NULL,
Address VARCHAR2(50)  NOT NULL,
Telno Number(15) NOT NULL,
CONSTRAINT college_telno_ck CHECK (Telno >= 1000000000),
CONSTRAINT college_name_uk UNIQUSQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL>
SQL> SQL> SP2-0734: unknown command beginning "Write DML ..." - rest of line ignored.
SQL> SQL> SP2-0734: unknown command beginning "1. Setup a..." - rest of line ignored.
SQL> SP2-0734: unknown command beginning "2. Now reg..." - rest of line ignored.
SQL> SQL>
Table dropped.

SQL> E(name),
CONSTRAINT college_id_pk PRIMARY KEY (id));

CREATE TABLE LOCATION (
id Number(5)  NOT NULL,
LocationName VARCHAR2(50) NOT NULL,
CONSTRAINT location_name_uk UNIQUE(LocationName),
CONSTRAINT location_id_pk PRIMARY KEY (id));

CREATE TABLE DEPARTMENT (
id Number(5) NOT NULL,
Name VARCHAR2(50) NOT NULL,
CollegeID Number(5) NOT NULL,
Locationid Number(5) NOT NULL,
CONSTRAINT department_name_uk UNIQUE(name),
CONSTRAINT department_id_pk PRIMARY KEY (id),
CONSTRAINT department_collegeid_fk FOREIGN KEY (CollegeID) REFERENCES
COLLEGE (id),
CONSTRAINT department_locationid_fk FOREIGN KEY (Locationid) REFERENCES
LOCATION (id));
```

```
CREATE TABLE COURSE (
id Number(5)  NOT NULL,
CourseNo Number(10)  NOT NULL,
CourseName VARCHAR2(50) NOT NULL,
Prereq VARCHAR2(50),
Deptid Number(5)  NOT NULL,
CONSTRAINT course_courseno_ck CHECK (CourseNo BETWEEN 10 AND 999),
CONSTRAINT course_name_uk UNIQUE(CourseName),
CONSTRAINT course_id_pk PRIMARY KEY (id),
CONSTRAINT course_deptid_fk FOREIGN KEY (Deptid) REFERE
Table dropped.

SQL> NCES DEPARTMENT (id));

CREATE TABLE RESIDENT (
Id Number(5) NOT NULL,
ResiName VARCHAR2(50) NOT NULL,
Feeschedule VARCHAR2(50) NOT NULL,
CONSTRAINT resident_feeschedule CHECK (Feeschedule IN ('Resident', 'NonResident')),
CONSTRAINT resident_id_pk PRIMARY KEY (id));

CREATE TABLE USERS (
id Number(5) NOT NULL,
Firstname VARCHAR2 (20) NOT NULL,
Lastname VARCHAR2 (20) NOT NULL,
Address VARCHAR2 (20) NOT NULL,
Email VARCHAR2 (20) NOT NULL,
Telno Number(15) NOT NULL,
ResidentId Number(5),
Type VARCHAR2(50) NOT NULL,
CONSTRAINT user_id_pk PRIMARY KEY (id),
CONSTRAINT user_residentid_fk FOREIGN KEY (ResidentId) REFERENCES
RESIDENT(id),
CONSTRAINT user_telno_ck CHECK (Telno >= 1000000000));

CREATE TABLE SECTION(
id Number(5) NOT NULL,
SectionNo Number(5) NOT NULL,
Courseid Number(5) NOT NULL,
Locationid Number(5) NOT NULL,
Starttime Date NOT NULL,
Endtime Date NOT NULL,
Startdate Date NOT NULL,
Enddate Date NOT NULL,
userid Number(5) NOT NULL,
CONSTRAINT section_id_pk PRIMARY KEY (id),
```

```
CONSTRAINT sect
Table dropped.

SQL>
Table dropped.

SQL> ion_courseid_fk FOREIGN KEY (Courseid) REFERENCES COURSE (id),
CONSTRAINT section_locationid_fk FOREIGN KEY (Locationid) REFERENCES
LOCATION (id),
CONSTRAINT section_userid_fk FOREIGN KEY (userid) REFERENCES USERS (id),
CONSTRAINT section_sectionno_ck CHECK (SectionNo IS NOT NULL AND SectionNo
BETWEEN 10 AND 999));

CREATE TABLE PAYMENT  (
Id Number(10) NOT NULL,
PaymentName VARCHAR2 (20),
CONSTRAINT payment_id_pk PRIMARY KEY (id));

CREATE TABLE STUDENTREGISTRATION (
id Number(5) NOT NULL,
StudentID Number(15) NOT NULL,
SectionId Number(5) NOT NULL,
FeeStatus VARCHAR2(50) NOT NULL,
Paymentid Number(10) NOT NULL,
CONSTRAINT SR_studentid_ck CHECK (StudentID IS NOT NULL AND StudentID >=
10000),
CONSTRAINT SR_id_pk PRIMARY KEY (id),
CONSTRAINT SR_studentid_fk FOREIGN KEY (StudentID) REFERENCES USERS (id),
CONSTRAINT SR_sectionid_fk FOREIGN KEY (SectionID) REFERENCES SECTION (id),
CONSTRAINT SR_paymentid_fk FOREIGN KEY (Paymentid) REFERENCES PAYMENT
(id));

CREATE VIEW CATALOG AS SELECT
colle
Table dropped.

SQL> ge.id,
department.Name,
course.CourseNo,
course.CourseName,
LocationName,
section.Starttime,
section.Endtime,
section.Startdate,
section.Enddate
FROM college
```

```
FULL JOIN department
ON college.id = department.CollegeId
FULL JOIN course
ON department.id = course.Deptid
FULL JOIN location
ON location.id = department.Locationid
FULL JOIN section
ON course.id = section.Courseid AND location.id = section.Locationid ;

COMMIT;

/* insert into report 1*/
INSERT INTO COLLEGE (id, Name, Address, Telno)
VALUES (1, 'Engineering', '3605 Abc Way, Sunnyvale, CA 94086', 16508761289);

INSERT INTO LOCATION (id, LocationName)
VALUES (2, 'North Campus');

INSERT INTO DEPARTMENT (id, Name, CollegeId, LocationI
Table dropped.

SQL> d)
VALUES (15, 'Compute
Table dropped.

SQL> r Science', 1, 2);

Table dropped.

SQL> INSERT INTO COURSE (
Table dropped.

SQL> id, CourseNo, Cours
View dropped.

SQL> SQL> e  2  Name, Prereq, DeptId)
V  3  ALUES (3, 101, 'Intro to  4   Java', NULL, 15);

INSERT I  5  NTO USERS (id, Firstname, Lastnam  6  e, Address, Email,
        Telno,   7  ResidentId, Type)
VALUES (62, 'Elizabeth', 'Ma',
        '216 Pr  8  ogress Ave.', 'lilyma@gmail.com',
        13457  9  263524, null, 'Professor');

INSERT INTO USE
Table created.
```

```
SQL> SQL> R  2  S
VALUES (17, 'Alex', '  3  Hornstein',
        '333 Flore  4  nce Road.', 'alexh@gmail.com',
        3107  5  458899, null, 'Professor');

INSERT INTO SECT ION (id, SectionNo, CourseId, Locationid)
Table created.

SQL> SQL> r  2  ttime, Endtime, Startdate,   3  Enddate, Userid)
VALUES   4  (5, 103, 3, 2,
        TO_DATE('18  5  :00:00', 'hh24:mi:ss'),
        TO_DA  6  TE('19:30:00', 'hh24:mi:ss'),
        TO_DATE('03/01/2018', 'mm/dd/yyyy'),
        TO_DA  8  TE('06/30/2018', 'mm/dd/yyyy'),
        62);

INSERT   9  INTO SECTION (id, SectionNo, CourseId, Locationid,
        Starttime, Endtime, Startdate,  10  Enddate, Userid)
VALUES (6, 104, 3, 2,
        TO_DATE('15:00:00', 'HH24:MI:SS'),
        TO_DATE('16
Table created.

SQL> SQL> :  2  30:00', 'HH24:MI:SS'),   3
        TO_DATE('03/02/2019', '  4  mm/dd/yyyy'),
        TO_DATE('06/29/20  5  19', 'mm/dd/yyyy'),
        17);

/* produce report 1*/
SET LI  7  NESIZE 200;

TTITLE 'Depart  8  ment Course Report'
COLUMN Name FORMAT A20
COLUMN "Start Time" FORM  9  AT A10
COLUMN "End Time" FORMAT A10
COLUMN Cou 10  rseName FORMAT A20

SELECT d.name, c.Course 11  Name,
     s.SectionNo, s.CourseId, s.Locationid,
     to_char(s.Starttime
Table created.

SQL> SQL> ,  2  'HH24:MI:SS') "Start Time  3  ",
     to_char(s.En  4  dtime, 'HH24:MI:SS') "End Time",  5
     s.Startdate "Start Date",   6
     s.Enddate "End Date",
```

```
      s.Userid "Professor ID",
      u.Firstname "P  7  ro First name",
      u.Lastname "Pro Last n
Table created.

SQL> SQL> a  2  me"
FROM DEPARTMENT d
  3  LEFT JOIN COURSE c
ON c  4  .DeptId = d.id
LEFT JOIN SECTION s  5
ON s.CourseId = c.id
LEFT JOIN U  6  SERS u
ON s.Userid = u.id;

/*    insert into report 2*/
INSERT   8  INTO RESIDENT (Id, ResiName,  9   Feeschedule)
VALUES  10  (20, 'Albert Branson', 'NonR 11  esident');

INSERT INTO USERS (id, First 12  name, Lastname, Address, Email,
          Telno, ResidentId, Type)
VALUES (23461, 'Alber 13  t', 'Branson',
          '499 Flower St.', 'branson@gmail.com',

Table created.

SQL> SQL>     2  13458762534, 20, 'Stude  3  nt');

INSERT INTO PAYM  4  ENT (id, PaymentName)
VALUES (  5  129, 'Visa');

INSERT INTO ST  6  UDENTREGISTRATION (id, StudentI  7  d, SectionId,
          FeeStatus  8  , PaymentId)
VALUES (66  9  1, 23461, 6, 'Paid', 129) 10  ;

/* produce report  2*/
SET LINESIZE 120;

TT 12  ITLE 'Student Registration Report'
COLUMN " 13  Fee Status" FORMAT A10

SELECT s.StudentId "Student ID",
      u.Firstname "F 14  irst name",
      u.Lastname "Last name",
      s.SectionId "Section ID",
      s 15  .FeeStatus "Fee Status",
      s.PaymentId "Payment ID",
```

```
      p.Paym 16  entName "Payment Name"
FROM STUDENTREGISTRATION s
LEFT JOIN PAYMENT p
ON s.PaymentId = p.id
LEFT
Table created.

SQL> SQL> J  2  OIN USERS u
ON s.Student  3  ID = u.id
LEFT JOIN RESI  4  DENT r
ON u.ResidentId = r.  5  id;

Table created.

SQL> SQL>   2    3    4    5    6    7    8    9   10   11   12
Table created.

SQL> SQL>   2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20
View created.

SQL> SQL>
Commit complete.

SQL> SQL> SQL>   2
1 row created.

SQL> SQL>   2
1 row created.

SQL> SQL>   2
1 row created.

SQL> SQL>   2
1 row created.

SQL> SQL>   2    3    4    5
1 row created.

SQL> SQL>   2    3    4
1 row created.

SQL> SQL>   2    3    4    5    6    7    8
1 row created.

SQL> SQL>   2    3    4    5    6    7    8
1 row created.
```

SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> 2 3 4 5
Fri Jun 19

Department Course Report

NAME COURSENAME SECTIONNO COURSEID LOCATIONID Start Time
End Time Start Date End Date Professor ID Pro First name Pro Last name
---------- ---------- ---------- ---------- ---------- ---------- ---------- --------- ---------
Computer Science Intro to Java 103 3 2 18:00:00 19:30:00 01-MAR-18
30-JUN-18 62 Elizabeth Ma
Computer Science Intro to Java 104 3 2 15:00:00 16:30:00 02-MAR-19
29-JUN-19 17 Alex Hornstein

SQL> SQL> SQL> 2
1 row created.
SQL> SQL> 2 3 4 5
1 row created.
SQL> SQL> 2
1 row created.
SQL> SQL> 2 3
1 row created.
SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> SQL> 2 3 4 5 6 7 8
Fri Jun 19

Student Registration Report

Student ID First name Last name Section ID Fee Status Payment ID Payment
Name
---------- ---------- ---------- ---------- ---------- ---------- ----------
23461 Albert Branson 6 Paid 129 Visa

SQL>

Recommendations

1, We can use range partitioning to the Payment table, because this table contains historical data (i.e. data is added every academic quarter), and old data can be stored in a data warehouse.

2, We can add an index for the Prereq column of the Course table, because this column is expected to have a lot of NULL values.

3, We can create an index for the composite (Starttime, Endtime) of the SECTION table, because these columns are often used together in WHERE clauses.

4, We can add an index to the SectionNo column of the SECTION table, because it is likely to be queried frequently with less than 2-4% of course sections returned on each query.