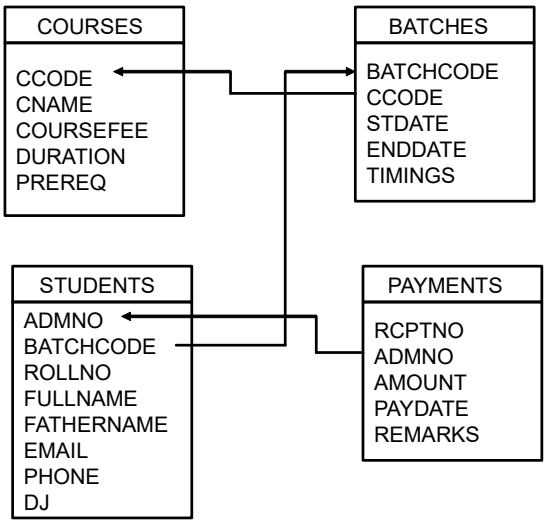
Students Database

The following commands are used to create a new user called STUDENT with password STUDENT, and grant required privileges to the user.

You must login in SYSTEM to execute the following commands.

```
Create user student identified by student;
Grant connect, resource to student;
Connect student/student;
```

The following tables are used to stored details of students of an institute.



Courses

This table contains details of all courses offered by the institute.

COLUMN NAME	DATATYPE	DESCRIPTION
CCODE	VARCHAR2(10)	COURSE CODE – PRIMARY KEY
CNAME	VARCHAR2(50)	COURSE NAME
COURSEFEE	NUMBER(6)	COURSE FEE IN INR.
DURATION	NUMBER(3)	DURATION OF COURSE IN MINUTES
PREREQ	VARCHAR2(100)	PREREQUISITE FOR THE COURSE

Batches

This table contains details of all batches – running and completed and yet to start.

COLUMN NAME	DATA TYPE	DESCRIPTION
BATCHCODE	VARCHAR2(10)	A UNIQUE CODE FOR EACH BATCH WITH FORMAT <COURSE><STDATE> - PRIMARY KEY
CCODE	VARCHAR2(10)	A FOREIGN KEY REFERENCING THE COURSE TAUGHT IN THIS BATCH.
STDATE	DATE	STARTING DATE OF THIS BATCH
ENDDATE	DATE	ENDING DATE OF THE BATCH. COULD BE NULL UNTIL BATCH IS COMPLETED
TIMINGS	VARCHAR2(20)	BATCH TIMINGS.

Students

This table contains details of all students of all batches. Each student is given a unique admission number. Each student in the batch is given a roll number, which is unique in the batch.

COLUMN NAME	DATA TYPE	DESCRIPTION
ADMNO	NUMBER(5)	A UNIQUE NUMBER ASSIGNED TO EACH STUDENT OF THE INSTITUTE – PRIMARY KEY
BATCHCODE	VARCHAR2(10)	BATCH CODE FOR THE BATCH TO WHICH STUDENT BELONGS. FOREIGN KEY, REFERENCES BATCHES TABLE.
ROLLNO	NUMBER(3)	ROLL NUMBER OF THE STUDENT IN THE BATCH
FULLNAME	VARCHAR2(50)	FULLNAME OF THE STUDENT.
FATHERNAME	VARCHAR2(50)	FATHERS NAME OF THE STUDENT
EMAIL	VARCHAR2(50)	EMAIL ADDRESS OF THE STUDENT.
PHONENO	VARCHAR2(20)	PHONE NUMBER OF THE STUDENT
DJ	DATE	DATE ON WHICH STUDENT JOINED.

Payments

This table contains details of all payments made by students.

COLUMN NAME	DATA TYPE	DESCRIPTION
RCPTNO	NUMBER(5)	RECEIPT NUMBER FOR PAYMENT – PRIMARY KEY.
ADMNO	NUMBER(5)	ADMISSION NUMBER OF THE STUDENT MAKING THE PAYMENT. FOREIGN KEY, REFERENCING STUDENTS TABLE.
AMOUNT	NUMBER(6)	AMOUNT PAID BY STUDENT
PAYDATE	DATE	DATE OF PAYMENT
REMARKS	VARCHAR2(200)	REMARKS REGARDING PAYMENT.

Commands to create tables and insert sample data

The following CREATE TABLE commands are used to create these tables. Each CREATE TABLE command is followed by some INSERT commands to insert sample data into table.

COURSES

```
CREATE TABLE COURSES
(
CCODE     VARCHAR2(10)  CONSTRAINT COURSES_PK  PRIMARY KEY,
CNAME     VARCHAR2(50)  CONSTRAINT COURSES_CNAME_NN NOT NULL,
COURSEFEE NUMBER(6),
DURATION  NUMBER(3),
PREREQ    VARCHAR2(100)
);
```

```
INSERT INTO COURSES VALUES('ORACLE11G','ORACLE DATABASE 11G',2500,40,'COMPUTER KONWLEDGE');
INSERT INTO COURSES VALUES('JAVASE6.0','JAVA SE 6.0',2500,40,'C LANGUAGE');
INSERT INTO COURSES VALUES('DOTNET3.5','MICROSOFT  .NET 3.5',3750,80,'C LANGUAGE & SQL');
```

BATCHES

```
CREATE TABLE BATCHES
(
BATCHCODE VARCHAR2(15)  CONSTRAINT BATCHES_PK PRIMARY KEY,
CCODE     VARCHAR2(10)  CONSTRAINT BATCHES_CCODE_FK  REFERENCES COURSES(CCODE),
STDATE    DATE,
ENDDATE   DATE,
TIMINGS   VARCHAR2(20),
CONSTRAINT BATCHES_DATE_CHECK  CHECK(STDATE <= ENDDATE)
);
```

```
INSERT INTO BATCHES VALUES('ORA130508','ORACLE11G','13-MAY-08','17-JUN-08','4:30 TO 6:00 PM');
INSERT INTO BATCHES VALUES('DOTNET130508','DOTNET3.5','13-MAY-08','26-JUN-08','7:00 TO 9:00
AM');
INSERT INTO BATCHES VALUES('ORA270608','ORACLE11G','27-JUN-08',NULL,'5:00 TO 6:00 PM'); INSERT
INTO BATCHES VALUES('JS270608','JAVASE6.0','27-JUN-08',NULL,'6:00 TO 7:00 PM');
```

STUDENTS

```
CREATE TABLE STUDENTS
(
ADMNO      NUMBER(5)    CONSTRAINT STUDENTS_PK  PRIMARY KEY,
BATCHCODE  VARCHAR2(15) CONSTRAINT STUDENTS_BATCHCODE_FK   REFERENCES  BATCHES (BATCHCODE),
ROLLNO     NUMBER(3) ,
FULLNAME   VARCHAR2(50) CONSTRAINT STUDENTS_FULLNAME_NN NOT NULL,
FATHERNAME VARCHAR2(50),
EMAIL      VARCHAR2(50),
PHONE      VARCHAR2(20),
DJ         DATE,
CONSTRAINT STUDENTS_BATCHCODE_ROLLNO_U UNIQUE(BATCHCODE,ROLLNO)
);
```

```
INSERT INTO STUDENTS VALUES(1,'ORA130508',1,'MICHEAL JORDON',
  'TIM JORDON','MJORDON@YAHOO.COM', '9873737334','11-MAY-08');
INSERT INTO STUDENTS VALUES(2,'ORA130508',2,'TIM SLIM',
  'TIM  KEN','TIM@YAHOO.COM', '9833334334','11-MAY-08');
INSERT INTO STUDENTS VALUES(3,'DOTNET130508',1,'HUNTER JASON',
  'HUNTER BOB','JHUNTER@YAHOO.COM', '34344343','11-MAY-08');
INSERT INTO STUDENTS VALUES(4,'JS270608',1,'JAMES GOODWILL',
  'JAMES ROBERTS','JAMES@YAHOO.COM', '9989898998','26-JUN-08');
INSERT INTO STUDENTS VALUES(5,'JS270608',2,'KENNY PETERSON',
  'KENNY JACOB','KPERERSON@GMAIL.COM', '9983373333','27-JUN-08');
INSERT INTO STUDENTS VALUES(6,'ORA270608',1,'GLEN JHONSON',
  'GLEN HENDRICK','GLEN@GMAIL.COM', '9898398985','28-JUN-08');
INSERT INTO STUDENTS VALUES(7,'ORA270608',2,'BATES KATHY',  'BATES
ROBERTS','KATHY@YMAIL.COM', '234423232','30-JUN-08');
```

PAYMENTS

```
CREATE TABLE PAYMENTS
(
RCPTNO    NUMBER(5)  CONSTRAINT PAYMENTS_PK PRIMARY KEY,
ADMNO     NUMBER(5)  CONSTRAINT PAYMENTS_ADMNO_FK
                          REFERENCES  STUDENTS(ADMNO)    ON DELETE CASCADE,
AMOUNT    NUMBER(6)  CONSTRAINT PAYMENTS_AMOUNT_NN NOT NULL
                     CONSTRAINT PAYMENTS_AMOUNT_CHECK  CHECK(AMOUNT > 0 ),
PAYDATE   DATE,
REMARKS   VARCHAR2(200)
);
```

```
INSERT INTO PAYMENTS VALUES(1,1,300,'11-MAY-08','REG. FEE');
INSERT INTO PAYMENTS VALUES(2,2,2500,'11-MAY-08','TOTAL FEE');
INSERT INTO PAYMENTS VALUES(3,3,1000,'11-MAY-08','REG. FEE');
INSERT INTO PAYMENTS VALUES(4,3,2750,'12-MAY-08',NULL);
INSERT INTO PAYMENTS VALUES(5,4,300,'26-JUN-08','REG. FEE');
INSERT INTO PAYMENTS VALUES(6,5,300,'27-JUN-08','REG. FEE');
INSERT INTO PAYMENTS VALUES(7,4,1700,'27-JUN-08',NULL);
INSERT INTO PAYMENTS VALUES(8,5,1700,'29-JUN-08',NULL);
INSERT INTO PAYMENTS VALUES(9,6,2500,'28-JUN-08','CHEQUE NO:3434343 SBI DWK');
INSERT INTO PAYMENTS VALUES(10,7,2500,'30-JUN-08',NULL);
```

Simple Queries

1. DISPLAY ALL STUDENTS IN THE ASCENDING ORDER OF BACHCODE AND JOINING DATE

```
SELECT * FROM STUDENTS ORDER BY BATCHCODE, DJ;
```

2. DISPLAY ALL PAYMENTS MADE IN THE MONTH OF MAY, 2008

```
SELECT * FROM PAYMENTS WHERE PAYDATE BETWEEN '1-MAY-08' AND '31-MAY-08';
```

3. DISPLAY ALL PAYMENT MADE THROUGH CHEQUE

```
SELECT * FROM PAYMENTS WHERE REMARKS LIKE '%CHEQUE%';
```

4. DISPLAY STUDENT NAME, FATHERNAME, JOINING DATE AND NO. OF DAYS SINCE JOINED.

```
SELECT FULLNAME, FATHERNAME, DJ, TRUNC( SYSDATE - DJ) NODAYS FROM
STUDENTS;
```

5. DISPLAY BATCHES THAT ARE CURRENTLY RUNNING.

```
SELECT * FROM BACHES WHERE ENDDATE IS NULL;
```

6. DISPLAY BATCHES OF JAVASE AND ORACLE.

```
SELECT * FROM BATCHES WHERE CCODE IN ('JAVASE6.0','ORACLE11G');
```

7. DISPLAY DUE DATE FOR THE PAYMENT ASSUMING DUE DATE IS 7 DAYS FROM DJ.

```
SELECT FULLNAME, BATCHCODE, DJ, DJ + 7 DUEDATE FROM STUDENTS;
```

8. DISPLAY DETAISL OF STUDENTS WHERE DUE DATE FOR PAYMENT IS OVER.

```
SELECT FULLNAME, BATCHCODE, DJ + 7 DUEDATE FROM STUDENTS WHERE  DJ + 7 <
SYSDATE
ORDER BY DJ + 7;
```

9. DISPLAY DETAILS OF COURSES WITH A PROPOSED INCREASE OF 10% IN COURSE FEE FOR COURSES WITH COURESE FEE LESS THAN 3000.

```
SELECT CNAME, COURSEFEE, COURSEFEE * 1.1 NEWFEE
FROM COURSES WHERE COURSEFEE < 3000;
```

10. DISPLAY STUDENTS WHOSE NAME CONTAINS LETTER 'S' AND FATHER'S NAME CONTAINS LETTER 'P'.

```
SELECT * FROM STUDENTS WHERE FULLNAME LIKE '%S%' AND FATERNAME LIKE '%P%';
```

11. DISPLAY BATCHES THAT ARE RUNNING FOR MORE THAN 45 DAYS.

```
SELECT * FROM BATCHES WHERE SYSDATE - STDATE > 45;
```

12. DISPLAY BATCHCODE, STDATE AND APROX. ENDING DATE FOR ORACLE BATCHES THAT ARE CURRENTLY RUNNING, IF BATCH TAKES TWO MONTHS.

```
SELECT BATCHCODE, STDATE, ADD_MONTHS(STDATE,2) APROXENDDATE FROM BATCHES
WHERE CCODE = 'ORACLE11G' AND ENDDATE IS NULL;
```

13. DISPLAY THE DIFFERENCE BEWTWEEN ACTUAL ENDING DATE AND ESTIMAED ENDDING DATE FOR JAVA BATHES ASSUMING EACH BATCH TAKES TWO MONTHS.

```
SELECT BATCHCODE, ENDDATE, ADD_MONTHS(STDATE,2) APOXENDDDATE,
ENDDATE - ADD_MONTHS(STDATE,2)  DAYSDIFFERENCE FROM BATCHES
WHERE CCODE = 'JAVASE6.0' AND ENDDATE IS NOT NULL;
```

14. INSERT COURSE DETAILS OF JAVA EE WEB COURSE.

```
INSERT INTO COURSES VALUES ('JAVAEEWEB','JAVA EE (WEB APPLICATIONS)',
3000,40,'JAVA LANG AND SQL');
```

15. UPDATE BATCHES TABLE TO SET ENDDATE OF BATCH JS130508 TO YESTERDAY.

```
UPDATE BATCHES SET ENDDATE = SYSDATE -1 WHERE BATCHCODE = 'JS130508';
```

16. DISPLAY BATCHES THAT STARTED IN THE PREVIOUS YEAR BUT ENDED IN THIS YEAR.

```
SELECT BATCHCODE FROM  BATCHES
WHERE TO_CHAR(STDATE,'YYYY') = TO_CHAR(SYSDATE,'YYYY') - 1
    AND TO_CHAR(ENDDATE,'YYYY') = TO_CHAR(SYSDATE,'YYYY');
```

17. DISPLAY PAYMENTS WITH AMOUNT MORE THAN 1000 OR MADE BY STUDENTS WITH ADMNO IN THE RANGE 100 AND 150 IN THE LAST 10 DAYS

```
SELECT * FROM PAYMENTS
WHERE ADMNO BETWEEN 100 AND 150 AND SYSDATE - PAYDATE <= 10 OR  AMOUNT > 1000;
```

18. CHANGE PAYDATE FOR RECEIPT 12 TO 1ST JUNE,2008 AND ADMNO TO 120.

```
UPDATE PAYMENTS SET ADMNO = 120, PAYDATE = '1-JUN-2008'
WHERE RCPTNO = 12;
```

19. DISPLAY STUDENT'S NAME, BATCHCODE, AND DJ IN ASCENDING ORDER OR NAME FOLLOWED BY DJ.

```
SELECT FULLNAME, BATCHCODE, DJ FROM STUDENTS
ORDER BY  FULLNAME, DJ;
```

20. DISPLAY APROX DATE WHEN CHEQUE WILL BE REALISED FOR CHEQUE PAYMENTS.

```
SELECT  RCPTNO, AMOUNT, PAYDATE, PAYDATE + 3 REALISATION_DATE FROM  PAYMENTS
WHERE REMARKS LIKE '%CHEQUE%';
```

GROUPING

1. DISPLAY TOTAL AMOUNT PAID BY ALL STUDENTS

```
SELECT SUM(AMOUNT) FROM PAYMENTS;
```

2. DISPLAY THE HIGHEST RECEIPT NUMBER FOR PAYMENTS IN THE MONTH OF MAY,2008.

```
SELECT MAX(RCPTNO)  FROM PAYMENTS
WHERE PAYDATE BETWEEN '1-MAY-08' AND '31-MAY-08';
```

3. DISPLAY BATCHCODE AND NO. OF STUDENTS IN THE BATCH.

```
SELECT BATCHCODE, COUNT(ROLLNO) FROM STUDENTS GROUP BY BATCHCODE;
```

4. DISPLAY THE MOST RECENTLY STATED BATCHED FOR EACH COURSE.

```
SELECT CCODE, MAX(STDATE)  FROM  BATCHES GROUP BY CCODE;
```

5. DISPLAY TOTAL AMOUNT PAID ON EACH DAY.

```
SELECT  TRUNC(PAYDATE), SUM(AMOUNT) FROM  PAYMENTS
GROUP BY TRUNC(PAYDATE);
```

6. DISPLAY NO. BATCHES FOR EACH COURSE IN THE CURRENT YEAR.

```
SELECT  CCODE, COUNT(*) NOBATCHES FROM  BATCHES
WHERE  TO_CHAR(STDATE,'YYYY') =  TO_CHAR(SYSDATE,'YYYY')
GROUP BY CCODE;
```

7. DISPLAY AMOUNT COLLECTED FOR EACH MONTH.

```
SELECT  TO_CHAR(PAYDATE,'MM-YYYY'), SUM(AMOUNT) FROM  PAYMENTS
GROUP BY  TO_CHAR(PAYDATE,'MM-YYYY');
```

8. DISPLAY BATCHES WHERE THE NUMBER OF STUDENTS IS MORE THAN 10.

```
SELECT  BATCHCODE, COUNT(ROLLNO) FROM  STUDENTS
GROUP BY BATCHCODE
HAVING COUNT(ROLLNO) > 10;
```

9. DISPLAY BATCHCODE AND FIRST AND LAST ADMISSION INTO BATCH.

```
SELECT BATCHCODE, MIN(DJ), MAX(DJ)
FROM STUDENTS
GROUP BY BATCHCODE;
```

10. DISPLAY COURSES FOR MORE THAN A BATCH WAS STARTED IN THE SAME MONTH.

```
SELECT CCODE FROM  BATCHES
GROUP BY  CCODE,TO_CHAR(STDATE,'MM-YYYY')
HAVING COUNT(*) > 1;
```

11. DISPLAY NO. OF BACHES FOR EACH YEAR AND COURSE.

```
SELECT  CCODE, TO_CHAR(STDATE,'YYYY'), COUNT(*) NOBATCHES FROM BATCHES
GROUP BY CCODE, TO_CHAR(STDATE,'YYYY');
```

12. USE ROLLUP AND CUBE TO DISPLAY BATHES FOR EACH YEAR AND COURSE.

```
SELECT  CCODE, TO_CHAR(STDATE,'YYYY'), COUNT(*) NOBATCHES FROM BATCHES
GROUP BY ROLLUP(CCODE, TO_CHAR(STDATE,'YYYY'));

SELECT  CCODE, TO_CHAR(STDATE,'YYYY'), COUNT(*) NOBATCHES FROM BATCHES
GROUP BY CUBE(CCODE, TO_CHAR(STDATE,'YYYY'))
ORDER BY CCODE,TO_CHAR(STDATE,'YYYY');
```

13. DISPLAY HOW MANY BACHES ARE CURRENTLY RUNNING.

```
SELECT COUNT(*) FROM BATCHES WHERE ENDDATE IS NULL;
```

14.DISPLAY NO. OF STUDENTS USING EACH MAIL SERVER.

```
SELECT  SUBSTR( EMAIL, INSTR(EMAIL,'@')+1), COUNT(*)
FROM STUDENTS
GROUP BY SUBSTR( EMAIL, INSTR(EMAIL,'@')+1);
```

15. DISPLAY NO. OF STUDENTS FOR EACH BATCH ON JAVA WITH MORE THAN 10 STUDENTS IN THE ASCENDING ORDER OF NO. OF STUDENTS

```
SELECT  BATCHCODE, COUNT(*) NOSTUDENTS  FROM STUDENTS
WHERE  BATCHCODE LIKE 'J%'
GROUP BY BATCHCODE
HAVING COUNT(*) >  10
ORDER BY NOSTUDENTS;
```

16. DISPLAY TOTAL AMOUNT RECEIVED FOR THE CURRENT MONTH.

```
SELECT SUM(AMOUNT) FROM PAYMENTS
WHERE TO_CHAR(SYSDATE,'MM-YY')  = TO_CHAR(PAYDATE,'MM-YY');
```

17. DISPLAY YEAR AND TOTAL PAYMENTS FOR THE YEAR.

```
SELECT  TO_CHAR(PAYDATE,'YYYY'), SUM(AMOUNT) FROM PAYMENTS
GROUP BY TO_CHAR(PAYDATE,'YYYY');
```

18. DISPLAY THE NUMBER OF COURSES STUDENTS WHERE NAME CONTAINS 'TOM' HAVE DONE.

```
SELECT  EMAIL, COUNT(*) NOCOURSES FROM STUDENTS
WHERE FULLNAME LIKE '%TOM%'
GROUP BY EMAIL;
```

19. DISPLAY DAYS ON WHICH MORE THAN 5000 WAS RECEIED AS PAYMENTS.

```
SELECT  TRUNC(PAYDATE), SUM(AMOUNT) FROM  PAYMENTS
GROUP BY TRUNC(PAYDATE)
HAVING SUM(AMOUNT) > 5000;
```

20. DISPLAY BATCHES WHERE NO. OF STUDENTS WHO JOINED AFTER IN THE LAST 10 DAYS ARE MORE THAN 5.

```
SELECT BATCHCODE, COUNT(*) NOSTUDENT
FROM STUDENTS
WHERE  SYSDATE - DJ <=10
GROUP BY BATCHCODE;
```

JOINING

1. DISPLAY BATCHCODE , COURSE NAME,STARTING DATE

```
SELECT  BATCHCODE, CNAME, STDATE FROM  BATCHES B, COURSES C
WHERE B.CCODE = C.CCODE;
```

2. DISPLAY RCPTNO, FULLNAME, AMOUNT PAID AND PAY DATE IN THE ORDER OF PAYDATE.

```
SELECT RCPTNO, FULLNAME, AMOUNT , PAYDATE FROM  PAYMENTS P, STUDENTS S
WHERE  P.ADMNO = S.ADMNO ORDER BY PAYDATE;
```

3. DISPLAY COURSE NAME, BATCHCODE AND FULLNAME.

```
SELECT CNAME, B.BATCHCODE, FULLNAME FROM STUDENTS S, BATCHES B, COURSES C
WHERE B.BATCHCODE = S.BATCHCODE AND C.CCODE = B.CCODE;
```

4. DISPLAY NO. OF STUDENTS JOINED FOR EACH COURSE.

```
SELECT CNAME, COUNT(*) FROM  STUDENTS S, BATCHES B, COURSES C
WHERE C.CCODE = B.CCODE AND B.BATCHCODE = S.BATCHCODE
GROUP BY CNAME;
```

5. DISPLAY THE AMOUNT PAID BY EACH STUDENT IN BATCH 'ORA130508'.

```
SELECT  ROLLNO,FULLNAME, SUM(AMOUNT) AMOUNTPAID
FROM  STUDENTS S, PAYMENTS P
WHERE S.ADMNO = P.ADMNO AND BATCHCODE = 'ORA130508'
GROUP BY  S.ROLLNO, FULLNAME ORDER BY ROLLNO
```

6. DISPLAY TOTAL AMOUNT PAID BY EACH STUDENT.

```
SELECT FULLNAME, SUM(AMOUNT) FROM PAYMENTS P, STUDENTS S
WHERE  S.ADMNO = P.ADMNO GROUP BY FULLNAME;
```

7. DISPLAY DETAILS OF BATCHES FOR COURSE WITH COURSE FEE MORE THAN 3000.

```
SELECT B.* FROM BATCHES B, COURSES C
WHERE C.CCODE = B.CCODE AND  COURSEFEE > 3000;
```

8. DISPLAY RCPTNO,FULLNAME,BATCHCODE,AMOUNT,PAYDATE FOR PAYMENTS IN THE LAST 10 DAYS.

```
SELECT  RCPTNO, FULLNAME, BATCHCODE, AMOUNT, PAYDATE FROM  STUDENTS S, PAYMENTS P
WHERE S.ADMNO = P.ADMNO AND  SYSDATE - PAYDATE <= 30
ORDER BY RCPTNO;
```

9. DISPLAY CNAME, BATCHCODE , STDATE AND ENDDATE FOR ALL BATCHES THAT ARE COMPLETED.

```
SELECT CNAME, BATCHCODE, STDATE, ENDDATE FROM BATCHES B, COURSES C
```

```
WHERE  B.CCODE = C.CCODE AND ENDDATE IS NOT NULL;
```

10. DISPLAY FULLNAME, DJ AND AMOUNTPAID AT THE TIME OF JOINING.

```
SELECT  FULLNAME, DJ, AMOUNT FROM  STUDENTS S, PAYMENTS P
WHERE  TRUNC(S.DJ) = TRUNC(P.PAYDATE) AND S.ADMNO = P.ADMNO;
```

11. DISPLAY COURSENAME, BATCHCODE INCLUDING COURSES THAT DO NOT HAVE ANY BATCHES.

```
SELECT  CNAME, BATCHCODE  FROM BATCHES B, COURSES C
WHERE  C.CCODE = B.CCODE (+);
```

12. DISPLAY NAMES OF THE STUDENTS WHO HAVE NOT PAID ANYTHING SO FAR.

```
SELECT FULLNAME FROM  STUDENTS S, PAYMENTS P
WHERE  S.ADMNO = P.ADMNO(+) AND P.AMOUNT IS NULL;
```

13. DISPLAY BATCHES THAT STARTED AFTER BATCH WITH CODE 'ORA130508'.

```
SELECT  B1.*
FROM BATCHES B1, BATCHES B2
WHERE B2.BATCHCODE = 'ORA130508' AND B1.STDATE > B2.STDATE;
```

14. DISPLAY FULLNAME, BATCHCODE FOR STUDENTS WHO HAVE PAID TOTAL AMOUNT AT THE TIME OF ADMISSION.

```
SELECT  FULLNAME, S.BATCHCODE, COURSEFEE
FROM   STUDENTS S, BATCHES B, PAYMENTS P, COURSES C
WHERE  S.ADMNO = P.ADMNO AND S.BATCHCODE = B.BATCHCODE AND C.CCODE = B.CCODE
AND  AMOUNT = COURSEFEE;
```

15. DISPLAY DETAILS OF STUDENTS WHO HAVE DUES.

```
SELECT  S.ADMNO,FULLNAME, S.BATCHCODE, COURSEFEE - SUM(AMOUNT) DUEAMOUNT
FROM STUDENTS S, PAYMENTS P, COURSES C, BATCHES B
WHERE S.ADMNO = P.ADMNO AND C.CCODE = B.CCODE AND B.BATCHCODE = S.BATCHCODE
GROUP BY S.ADMNO,FULLNAME, S.BATCHCODE, COURSEFEE
HAVING  COURSEFEE - SUM(AMOUNT) > 0
ORDER BY S.ADMNO;
```

SUB QUERIES

1. DISPLAY THE PAYMENTS MADE BY STUDENT 'JAMES GOODWILL'

```
SELECT * FROM PAYMENTS
WHERE ADMNO = (SELECT ADMNO FROM STUDENTS WHERE FULLNAME = 'JAMES GOODWILL');
```

2. DISPLAY PAYMENTS MADE BY STUDENTS WHO JOINED INTO 'ORA270608' BATCH.

```
SELECT * FROM PAYMENTS
WHERE ADMNO IN (SELECT ADMNO FROM STUDENTS WHERE BATCHCODE = 'ORA270608')
ORDER BY ADMNO;
```

3. DISPLAY BATCHES FOR COURSE WITH DURATION MORE THAN 40 HOURS.

```
SELECT * FROM BATCHES
WHERE CCODE IN ( SELECT CCODE FROM COURSES WHERE DURATION > 40);
```

4. DISPLAY STUDENTS WHO MADE PAYMENTS IN THE CURRENT MONTH.

```
SELECT * FROM STUDENTS WHERE ADMNO IN
(SELECT ADMNO FROM PAYMENTS
 WHERE TO_CHAR(SYSDATE,'MM-YYYY')=TO_CHAR(PAYDATE,'MM-YYYY')
 );
```

5. DISPLAY STUDENTS WHO JOINED INTO ORACLE COURSE.

```
SELECT FULLNAME  FROM STUDENTS
WHERE BATCHCODE IN ( SELECT BATCHCODE FROM BATCHES WHERE CCODE = 'ORACLE11G')
ORDER BY FULLNAME;
```

6. DISPLAY DETAILS OF STUDENTS FROM CURRENTLY RUNNING BATCHES.

```
SELECT * FROM STUDENTS WHERE BATCHCODE IN
        (SELECT BATCHCODE FROM BATCHES WHERE ENDDATE IS NULL);
```

7. DISPLAY FULLNAME, EMAIL ADDRESS OF ALL STUDENTS WHO COMPLETED BATCH 6 MONTHS BACK.

```
SELECT FULLNAME, EMAIL FROM  STUDENTS WHERE BATCHCODE IN
 (SELECT BATCHCODE FROM BATCHES WHERE  MONTHS_BETWEEN(SYSDATE,ENDDATE) > 6);
```

8. DISPLAY FULLNAME,EMAIL ADDRESS OF .NET STUDENTS WHO COMPLETED BATCH 6 MONTHS BACK.

```
SELECT FULLNAME, EMAIL FROM  STUDENTS WHERE BATCHCODE IN
 (SELECT BATCHCODE FROM BATCHES WHERE  MONTHS_BETWEEN(SYSDATE,ENDDATE) > 6
  AND  CCODE = 'DOTNET3.5')
```

9. DISPLAY DETAILS OF STUDENTS WHO BELONGED TO BATCH WITH LESS THAN 10 STUDENTS.

```
SELECT * FROM STUDENTS WHERE BATCHCODE IN
 ( SELECT BATCHCODE FROM STUDENTS GROUP BY BATCHCODE HAVING COUNT(*) < 10);
```

10. DISPLAY RCPTNO,FULLNAME,BATCHCODE, AMOUNT, PAYDATE FOR STUDENTS WHO JOINED INTO ORACLE COURSE.

```
SELECT RCPTNO, FULLNAME, BATCHCODE, AMOUNT, PAYDATE FROM STUDENTS S, PAYMENTS P
WHERE S.ADMNO = P.ADMNO AND BATCHCODE IN
   (SELECT BATCHCODE FROM BATCHES WHERE CCODE = 'ORACLE11G');
```

11. DISPLAY COURSE FOR WHICH WE HAVE ANY BATCH WITH MORE THAN 10 STUDENTS.

```
SELECT * FROM COURSES
WHERE CCODE IN
 (SELECT CCODE FROM BATCHES WHERE BATCHCODE IN
       (SELECT BATCHCODE FROM STUDENTS
        GROUP BY BATCHCODE HAVING COUNT(*) > 10)
 );
```

12. DISPLAY BATCHES FOR WHICH THE TOTAL AMOUNT COLLECTED IS MORE THAN 20000.

```
SELECT BATCHCODE, TOTALAMOUNT FROM
 (SELECT BATCHCODE, SUM(AMOUNT) TOTALAMOUNT
  FROM STUDENTS S, PAYMENTS P
  WHERE S.ADMNO = P.ADMNO
  GROUP BY BATCHCODE
  HAVING SUM(AMOUNT) > 20000);
```

13. DISPLAY STUDENTS WHO DID NOT JOIN ON THE DATE OF STARTING OF THE BATCH.

```
SELECT BATCHCODE, FULLNAME
FROM STUDENTS S WHERE DJ =
  ( SELECT STDATE FROM BATCHES WHERE BATCHCODE = S.BATCHCODE);
```

14. DISPLAY BATCHES WITH TOP 3 HIGHEST NO. OF STUDENTS

```
SELECT * FROM
  ( SELECT BATCHCODE,COUNT(*) FROM STUDENTS GROUP BY BATCHCODE
    ORDER BY 2 DESC)
WHERE ROWNUM < 4;
```

15. UPDATE AMOUNT IN RECEIPT 200 WITH TOTAL AMOUNT FOR COURSE IN WHICH STUDENT JOINED.

```
UPDATE PAYMENTS P
  SET AMOUNT = ( SELECT COURSEFEE FROM COURSES C, BATCHES B, STUDENTS S
                 WHERE C.CCODE = B.CCODE AND B.BATCHCODE = S.BATCHCODE
                 AND S.ADMNO = P.ADMNO);
WHERE RCPTNO = 2;
```