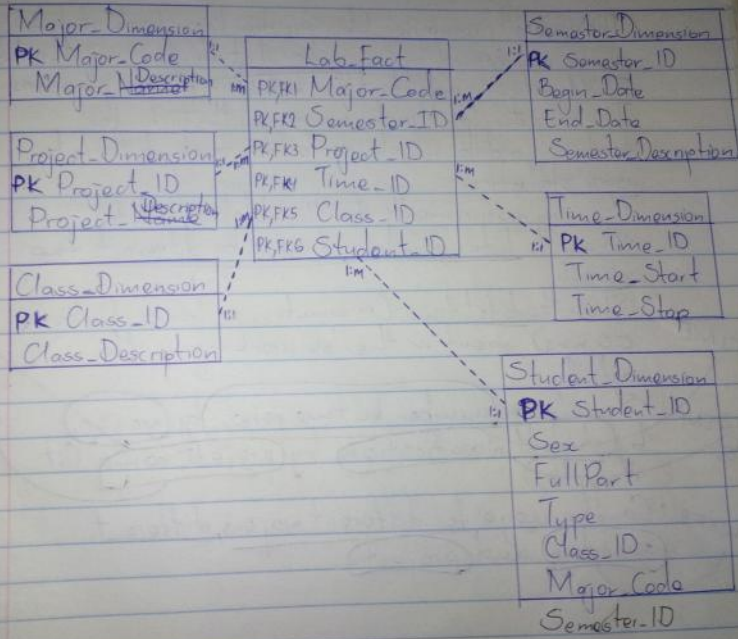


①



CREMENT,

The university computer lab's director keeps track of lab usage, as measured by the number of students using the lab. This function is important for budgeting purposes. The computer lab director assigns you the task for developing a data warehouse to keep track of the lab usage statistics.

The main requirements for this database are to:

- Show the total number of users by different time periods.
- Show the total time (in minutes, or hours if above 60 mins) spent in the lab working on the activity.
- Show usage number by time period, by major, by student classification, by project name (if applicable).
- Compare usage for different majors, different semesters and projects.

TIME

STUDENT
CLASS

SEMESTER

```
(student) => CREATE TABLE STUDENT_USAGE
(
  STUDENT_ID VARCHAR(10),
  SEX VARCHAR(1),
  FULL_PART VARCHAR(10),
  TYPE VARCHAR(1),
  CLASS_ID VARCHAR(3),
  MAJOR_CODE VARCHAR(3),
  PRIMARY KEY (STUDENT_ID)
);
```

CREATE DATABASE LABINFO;

USE LABINFO;

(Major) ⇒ CREATE TABLE MAJOR_DIM(
MAJOR_CODE ~~INT~~ VARCHAR(5) NOT NULL,
MAJOR_~~NAME~~^{DESCRIPTION} VARCHAR(40),
PRIMARY KEY(MAJOR_CODE)
);

(Semester) ⇒ CREATE TABLE SEMESTER_DIM(
SEMESTER_ID VARCHAR(5) NOT NULL,
SEMESTER_DESCRIPTION VARCHAR(15),
BEGIN-DATE DATE,
END-DATE DATE,
PRIMARY KEY(SEMESTER_ID)
);

(Project) ⇒ CREATE TABLE PROJECT_DIM(
PROJECT_ID INT NOT NULL AUTO_INCREMENT,
PROJECT_DESCRIPTION VARCHAR(40),
PRIMARY KEY(PROJECT_ID)
);

(Time) ⇒ CREATE TABLE TIME_DIM(
TIME_ID INT NOT NULL AUTO_INCREMENT,
TIME_START TIME,
TIME_STOP TIME,
PRIMARY KEY(TIME_ID)
);

(Class) ⇒ CREATE TABLE CLASS_DIM(
CLASS_ID VARCHAR(5) NOT NULL,
CLASS_DESCRIPTION VARCHAR(40),
PRIMARY KEY(CLASS_ID)
);

(Student) ⇒ CREATE TABLE STUDENT_DIM(
STUDENT_ID VARCHAR(20) NOT NULL,
SEX VARCHAR(1),
FULL_PART VARCHAR(1),
TYPE VARCHAR(1),
CLASS_ID VARCHAR(5),
MAJOR_CODE VARCHAR(5),
PRIMARY KEY(STUDENT_ID)
);

```
INSERT INTO MAJOR_DIM(MAJOR_CODE, MAJOR_DESCRIPTION)
SELECT DISTINCT MJ.MAJOR_CODE, MJ.MAJOR_DESCRIPTION
FROM production.STUDENT AS ST, production.MAJOR as MJ
WHERE
ST.MAJOR_CODE = MJ.MAJOR_CODE;
```

```
INSERT INTO SEMESTER_DIM(SEMESTER_ID, SEMESTER_DESCRIPTION,
BEGIN_DATE, END_DATE)
SELECT DISTINCT MJ.SEMESTER_ID, MJ.SEMESTER_DESCRIPTION,
MJ.BEGIN_DATE, MJ.END_DATE
FROM production.STUDENT AS ST, production
```

```
INSERT INTO LABINFO.SEMESTER_DIM(SEMESTER_ID,
SEMESTER_DESCRIPTION, BEGIN_DATE, END_DATE)
SELECT SEMESTER_ID, SEMESTER_DESCRIPTION, BEGIN_DATE,
END_DATE
FROM production.SEMESTER
```

```
INSERT INTO LABINFO.PROJECT_DIM(PROJECT_ID, PROJECT_DESCRIPTION)
PRIMARY SELECT PROJECT_ID, PROJECT_DESCRIPTION
FROM production.GROUP_PROJECTS.
```

```
INSERT INTO LABINFO.TIME_DIM(TIME_START, TIME_STOP)
SELECT TIME_START, TIME_STOP
FROM production.TIME
```

```
INSERT INTO LABINFO.CLASS_DIM(CLASS_ID, CLASS_DESCRIPTION)
SELECT CLASS_ID, CLASS_DESCRIPTION
FROM production.CLASS
```

PTION)
PTION
s MJ

```
INSERT INTO LABINFO.STUDENT_DIM(STUDENT_ID, SEX, FULLPART,  
TYPE, CLASS_ID, MAJOR_CODE)  
SELECT FROM STUDENT_ID, SEX, FULLPART, TYPE, CLASS_ID, MAJOR_CODE  
FROM production.STUDENT
```

SCRIPTION,

PTION,

iii. ~~SELECT~~ BEGIN-DATE, END-DATE FROM ^{SEMESTER_DIM} MAJOR_DIM
WHERE

③

②

i. SELECT * FROM MAJOR_DIM;

ii. SELECT BEGIN-DATE, END-DATE FROM SEMESTER_DIM;

iii. SELECT STUDENT-ID, MAJOR_CODE
FROM STUDENT
WHERE SEX = 'F';

iv. SELECT MAJOR_DESCRIPTION
FROM MAJOR_DIM

~~v. SELECT STUDENT-ID, MAJOR_DESCRIPTION~~

v. SELECT STUDENT_DIM.STUDENT-ID, MAJOR_DIM.MAJOR-ID,
MAJOR_DIM.MAJOR_DESCRIPTION
FROM STUDENT_DIM
INNER JOIN MAJOR_DIM ON STUDENT_DIM.MAJOR_CODE =
MAJOR_DIM.MAJOR_CODE;

2. CREATE VIEW v_List_Majors
(a) AS

```
SELECT MAJOR_DESCRIPTION  
FROM MAJOR_DIM  
ORDER BY MAJOR_DESCRIPTION DESC;
```

ii. CREATE VIEW v_Class

AS

```
SELECT DISTINCT CLASS_DIM.CLASS_DESCRIPTION,  
SUM(TOTAL_VISITS) AS TOTAL_VISITS, SUM(TOTAL_TIME)  
AS TOTAL_MINUTES, SUM(TOTAL_TIME)/60 AS TOT_HOURS,  
(SUM(TOTAL_TIME)/60)/SUM(TOTAL_VISITS) AS AVG_TIME  
FROM LAB_FACT AS LF, MAJOR_DIM, CLASS_DIM,  
PROJECTS_DIM  
WHERE LF.PROJECT_ID = PROJECTS_DIM.PROJECT_ID AND  
LF.MAJOR_CODE = MAJOR_DIM.MAJOR_CODE AND  
LF.CLASS_ID = CLASS_DIM.CLASS_ID  
GROUP BY CLASS_DIM.CLASS_DESCRIPTION.
```

⑥ DELETE FROM PROJECTS_DIM
WHERE PROJECT_ID = "2"

INSERT INTO PROJECT_DIM(PROJECT_DESCRIPTION)
VALUES("Final Project");

ii) DELETE FROM PROJECT_DIM
WHERE PROJECT_ID <= 3;

INSERT INTO MAJOR_DIM(MAJOR_CODE, MAJOR_DESCRIPTION)
VALUES("Psychology", "MND33")

iii) INSERT INTO CLASS_DIM(CLASS_ID, CLASS_DESCRIPTION)
SELECT DISTINCT STUDENT.CLASS_ID, CLASS_DIM.CLASS_DESCRIPTION
FROM production.STUDENT ~~AS ST~~, CLASS_DIM
WHERE
STUDENT.CLASS_ID = CLASS_DIM.CLASS_ID;

2. CREATE VIEW ^{v.l}
a) AS

SELECT MAJOR_DESCRIPTION
FROM MAJOR_DIM
ORDER BY MAJOR_CODE

ii. CREATE VIEW ^{v.l}
AS

SELECT DISTINCT
SUM(TOTAL_COST) AS TOTAL_COST
(SUM(TOTAL_COST))
FROM LAB_FACT
PROJECTS_DIM
WHERE LF.PROJECT_ID =
LF.MAJOR_CODE
LF.CLASS_ID
GROUP BY