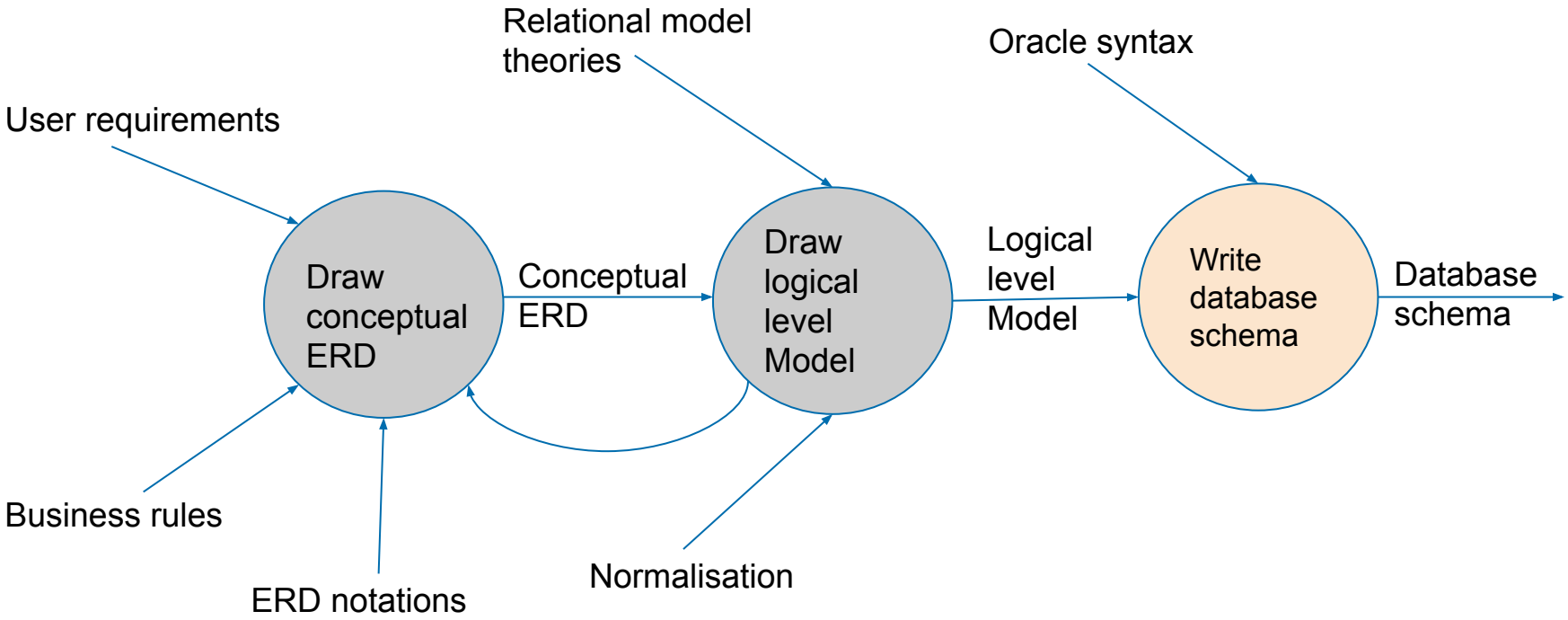
MONASH University

MONASH
INFORMATION
TECHNOLOGY

Creating & Populating the Database – Data Definition Language


Relational model theories
Oracle syntax
User requirements
Draw conceptual ERD
Conceptual ERD
Draw logical level Model
Logical level Model
Write database schema
Database schema
Business rules
ERD notations
Normalisation

SQL general syntax

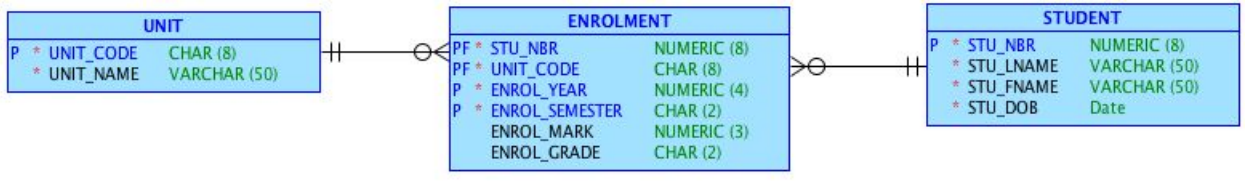
- A single statement is ended with SEMICOLON.
- Predefined KEYWORDs represent clauses (components) of a statement.
- Keywords are NOT case sensitive.
- Examples:

```
CREATE TABLE unit
  (
    unit_code  CHAR(7)NOT NULL,
    unit_name  VARCHAR2(50)CONSTRAINT uq_unit_name UNIQUE NOT NULL,
    CONSTRAINT pk_unit PRIMARY KEY (unit_code)
  );
SELECT * FROM student;
```

SQL Statements

- Data Definition Language (DDL)
  - Creating database structure
    - CREATE TABLE, ALTER TABLE, DROP TABLE
- Data Manipulation Language (DML)
  - Adding and Manipulating database contents (rows)
    - INSERT, UPDATE, DELETE
  - Retrieving data from database
    - SELECT
- Data Control Language (DCL)
  - Set permissions on objects
    - GRANT

Q1. There are a number of business rule represented by the above model. Choose true statement(s) according to the diagram.



A. A student enrols in a maximum of one unit.

B. An enrolment record is created for a particular student of a unit in a given semester and year.

C. A student can have more than one grade for a given unit.

D. A unit can only have a single student enrolled.

E. More than one option in a to d is correct.

CREATE A TABLE (DDL)

Q2. What relational model component(s) is/are defined in this create table statement?

```
CREATE TABLE STUDENT (
      stu_nbr       NUMBER(6)       NOT NULL,
      stud_lname    VARCHAR2(50)    NOT NULL,
      stud_fname    VARCHAR2(50)    NOT NULL,
      stu_dob       DATE            NOT NULL,
      CONSTRAINT STUDENT_PK PRIMARY KEY (stu_nbr)
);
```

A. Relation, Attribute, Domain

B. Primary Key

C. Foreign Key

D. Referential Integrity constraint

E. All of the options in a-d are correct.

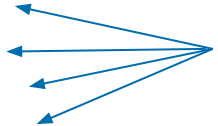
F. Some of the options in a-d are correct.

Common ORACLE data types

- **Text:** CHAR(size), VARCHAR2(size)
  - e.g., CHAR(10), VARCHAR2(10)
  - CHAR(10) → ‘apple’ = ‘apple ‘
  - VARCHAR2(10) → ‘apple’ != ‘apple ‘
- **Numbers:** NUMBER(precision, scale)
  - Weight NUMBER(7) or NUMBER(7,0) → Weight = 7456124
  - Weight NUMBER(9,2) → Weight = 7456123.89
  - Weight NUMBER(8,1) → Weight = 7456123.9
- **Data/Time:** DATE, TIMESTAMP
  - DATE can store a date and time (time to seconds), stored as Julian date
  - TIMESTAMP can store a date and a time (up to fractions of a second)
  - TIMESTAMP WITH TIME ZONE

Column VS Table Level Constraints

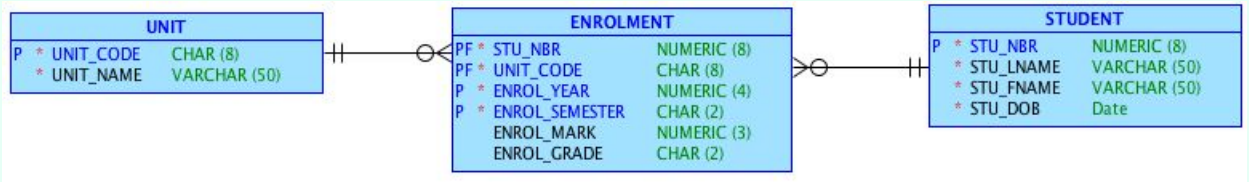
```
CREATE TABLE STUDENT (
        stu_nbr   NUMBER(6)  NOT NULL,
        stud_lname  VARCHAR2(50) NOT NULL,
        stud_fname  VARCHAR2(50)  NOT NULL,
        stu_dob   DATE NOT NULL,
        CONSTRAINT STUDENT_PK PRIMARY KEY (stu_nbr)
);
```



column constraints

table constraint

Q3. What would be the order of the CREATE TABLE statements in the schema script to successfully create a database based on the below diagram? (assuming that we will define the FK as part of the create table statement)



A. UNIT, ENROLMENT, STUDENT

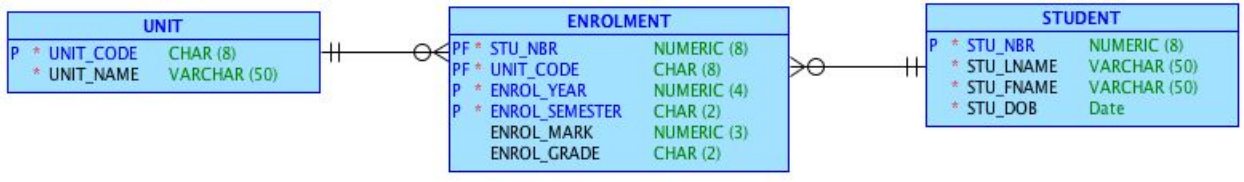
B. ENROLMENT, STUDENT, UNIT

C. STUDENT, UNIT, ENROLMENT

D. UNIT, STUDENT, ENROLMENT

E. More than one option is correct

Q4. How many foreign key/s (FK) will be in the database when the three tables are created?



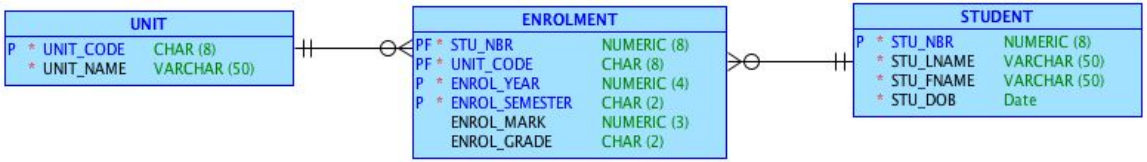
A. 1.

B. 2.

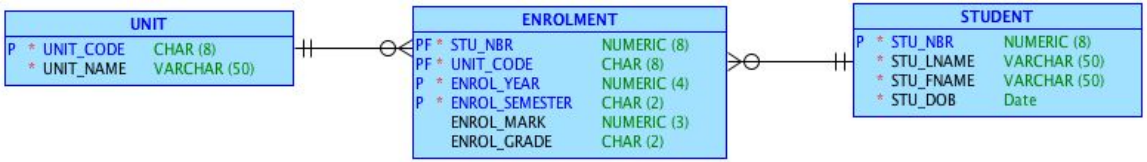
C. 3.

D. 4.

During answering, identify the attribute(s) that will be assigned as FK and what table(s) would it "link"?




```
CREATE TABLE student
  (
    stu_nbr         NUMBER(8)      NOT NULL,
    stu_lname       VARCHAR(50)    NOT NULL,
    stu_fname       VARCHAR(50)    NOT NULL,
    stu_dob         DATE           NOT NULL,
    CONSTRAINT pk_student PRIMARY KEY (stu_nbr)
  );

CREATE TABLE unit
  (
    unit_code       CHAR(8)          NOT NULL,
    unit_name       VARCHAR(50) CONSTRAINT uq_unit_name UNIQUE NOT NULL ,
    CONSTRAINT pk_unit PRIMARY KEY (unit_code)
  );
```




```
CREATE
  TABLE enrolment
  (
    stu_nbr              NUMBER(8)     NOT NULL,
    unit_code            CHAR(8)       NOT NULL,
    enrol_year           NUMBER(4)     NOT NULL,
    enrol_semester            CHAR(2)          NOT NULL,
    enrol_mark                NUMBER(3) ,
    enrol_grade               CHAR(2),
    CONSTRAINT pk_enrolment PRIMARY KEY (stu_nbr, unit_code, enrol_year, enrol_semester),
    CONSTRAINT fk_enrolment_student FOREIGN KEY (stu_nbr) REFERENCES student (stu_nbr),
    CONSTRAINT fk_enrolment_unit FOREIGN KEY (unit_code) REFERENCES unit (unit_code)
  );
```

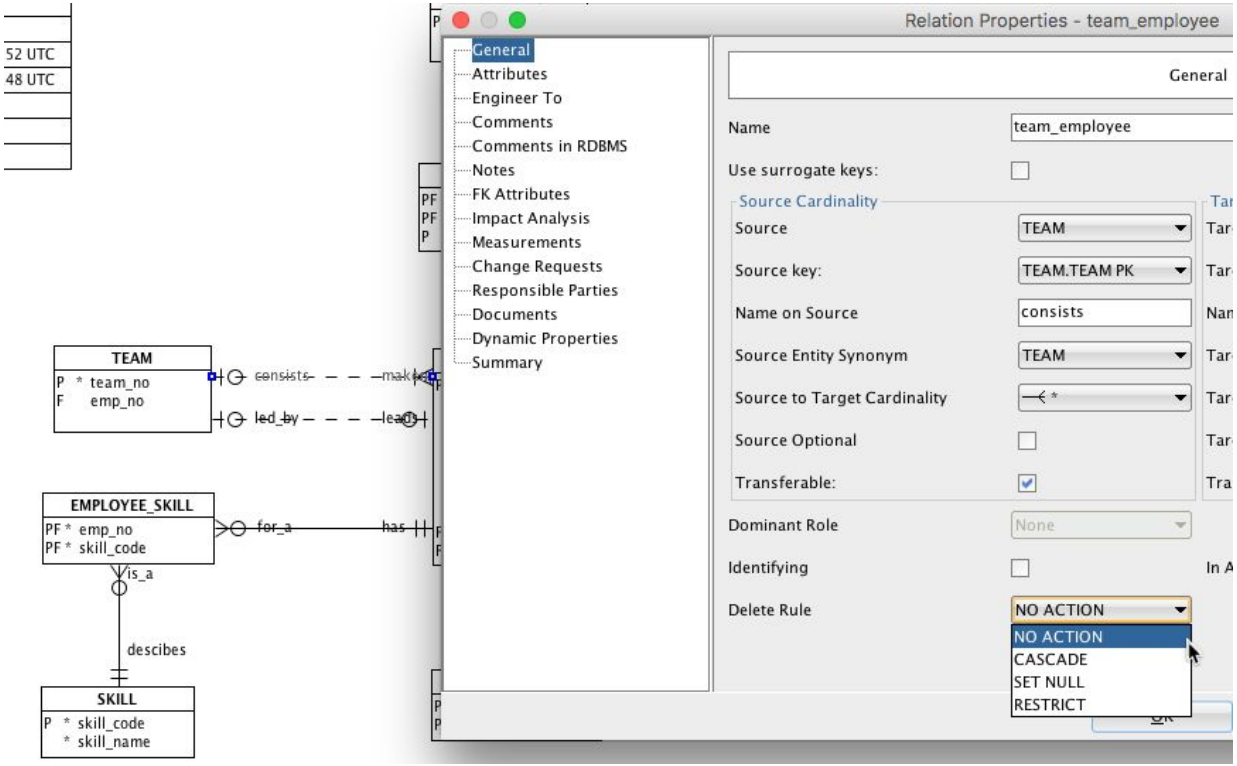
Alternative method of defining FKs

```
CREATE TABLE enrolment
  (
    stu_nbr              NUMBER(8)      NOT NULL,
    unit_code            CHAR(8)        NOT NULL,
    enrol_year           NUMBER(4)      NOT NULL,
    enrol_semester       CHAR(2)        NOT NULL,
    mark                 NUMBER(3),
    grade                CHAR(2),
    CONSTRAINT pk_enrolment PRIMARY KEY  (stu_nbr, unit_code, enrol_year, enrol_semester)
  );

ALTER TABLE enrolment
    ADD
          ( CONSTRAINT fk_enrolment_student FOREIGN KEY (stu_nbr)
                                  REFERENCES student ( stu_nbr),
            CONSTRAINT fk_enrolment_unit FOREIGN KEY (unit_code)
                                  REFERENCES unit  (unit_code));
```

Referential Integrity

- To ensure referential integrity, SQL defines three possible actions for FKs in relations when a deletion of a primary key occurs:
  - RESTRICT (Oracle No Action basically equivalent)
    - Deletion of tuples is NOT ALLOWED for those tuples in the table referred by the FK (the table containing PK) if there is corresponding tuple in the table containing the FK.
  - CASCADE
    - A deletion of a tuple in the table referred by the FK (the table containing PK) will result in the deletion of the corresponding tuples in the table containing the FK.
  - NULLIFY
    - A deletion of a tuple in the table referred by the FK (the table containing PK) will result in the update of the corresponding tuples in the table containing the FK to NULL.

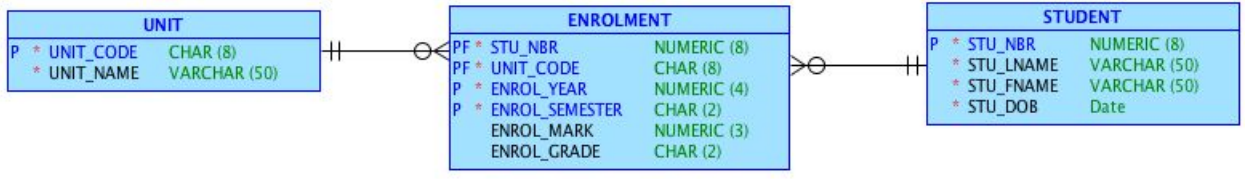
Referential Constraints SQL Data Modeller



What Referential Integrity Constraint to implement?

- Use the model to decide on what referential integrity constraint to implement.
  - Mandatory vs Optional participation.
- The constraints must be decided at the design phase.

Q5. Assume that the table ENROLMENT contains enrolment details for students in FIT9132 and FIT9001. The referential integrity constraint is CASCADE. What would happen to tuples in ENROLMENT with the unit_code='FIT9132' when we delete the FIT9132 record from UNIT?



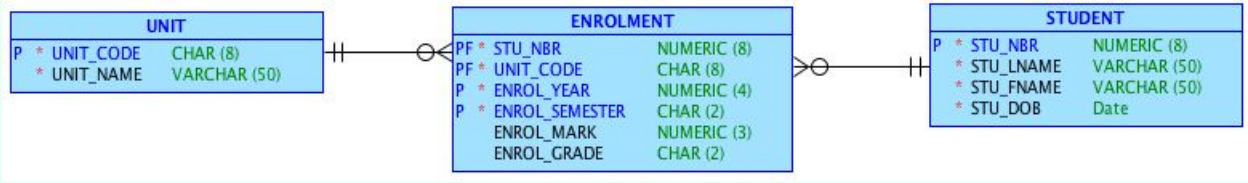
A. They will be deleted.

B. The value of unit_code will be updated to NULL.

C. The deletion is not possible, the DBMS will prevent the deletion.

D. None of the above.

Q6. What would happen to the student record with stu_nbr='1234' in the STUDENT table when we delete all tuples with stu_nbr='1234' in the ENROLMENT table? (Assume referential integrity is CASCADE constraints)



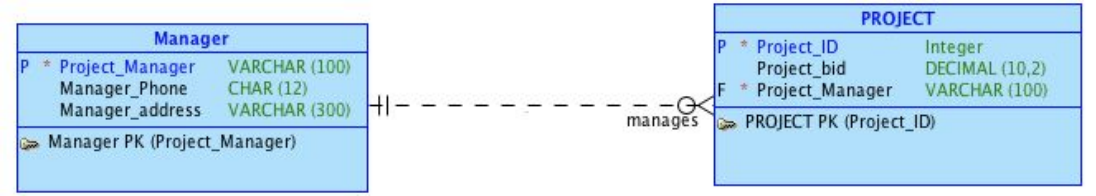
A. Student record with stu_nbr='1234' in the STUDENT table will be deleted.

B. Nothing will happen to the STUDENT table.

C. The stu_nbr='1234' in the STUDENT table will be updated to NULL.

D. Deletion will not be permitted by the DBMS.

Q7. What referential integrity constraint could be implemented according to the above model for the FKs in the PROJECT table without violating the business rules depicted in the model?



A. NULLIFY
B. CASCADE
C. RESTRICT
D. b and c are correct.
E. a, b and c are correct.

ALTER TABLE

- Used to change a tables structure.
- For example:
  - Adding column(s).
  - Removing column(s).
  - Adding constraint(s).
  - Removing constraint(s)

```
ALTER TABLE student
   ADD (stu_address varchar(200),
        status char(1) DEFAULT  'C',
        constraint status_chk CHECK (status in ('G','C'))
        );
```

Referential Integrity Definition - Example

```
ALTER TABLE enrolment
      DROP CONSTRAINT fk_enrolment_student;

ALTER TABLE enrolment
      DROP CONSTRAINT fk_enrolment_unit;

ALTER TABLE enrolment
   ADD
         ( CONSTRAINT fk_enrolment_student FOREIGN KEY (stu_nbr)
         REFERENCES student ( stu_nbr) ON DELETE CASCADE,

         CONSTRAINT fk_enrolment_unit FOREIGN KEY (unit_code) REFERENCES unit
                  (unit_code) ON DELETE CASCADE
          );
```

DELETING A TABLE

- Use the DROP statement.
- Examples:
  - `DROP TABLE enrolment PURGE;`
  - `DROP TABLE student CASCADE CONSTRAINTS PURGE;`

ADDING TUPLES/ROWS TO A TABLE (DML)

INSERT

- Adding data to a table in a database.
- SYNTAX:

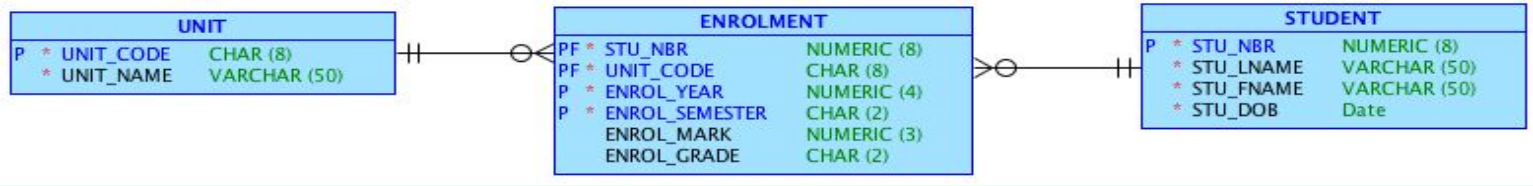
INSERT INTO *table [(column [, column...])]*

VALUES *(value [, value...]);*

```
INSERT INTO unit VALUES ('FIT9132', 'Databases');
INSERT INTO student VALUES (112233, 'Wild','Wilbur',
                            '01-Jan-1995')
```

Role of: to_date and to_char

Q8. Assume the tables have been created with primary and foreign key constraints and there is no data currently in the tables. In what order should we populate the table?



A. UNIT- > ENROLMENT -> STUDENT

B. STUDENT -> ENROLMENT -> UNIT

C. STUDENT -> UNIT -> ENROLMENT

D. More than one option is correct.

COMMIT and ROLLBACK

```
INSERT INTO enrolment VALUES (112233,
                  'FIT9132',1,2018,45,'N');
INSERT INTO enrolment VALUES (112233,
                  'FIT9001',1,2018,80,'HD');
COMMIT;
```

COMMIT makes the changes to the database permanent.

ROLLBACK will undo the changes.

Using a SEQUENCE

- Oracle supports auto-increment of a numeric PRIMARY KEY.
  - SEQUENCE.
- Steps to use:
  - Create sequence

```
CREATE SEQUENCE sno_seq
INCREMENT BY 1;
```

  - Access the sequence using two built-in variables (pseudocolumns):
    - NEXTVAL and CURRVAL

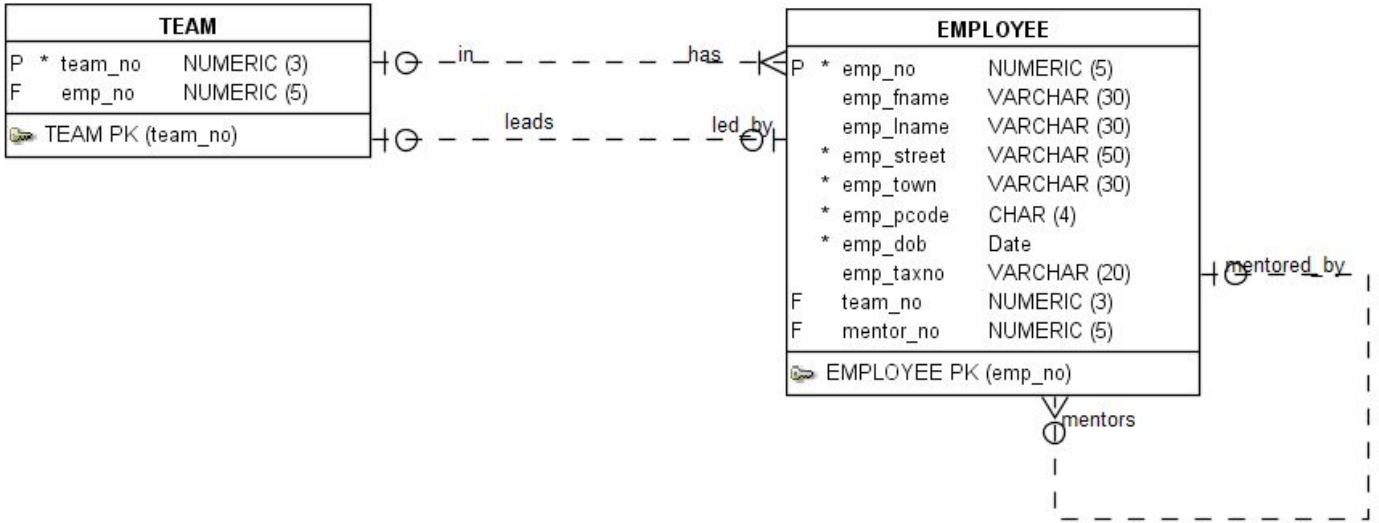
```
- INSERT INTO student
  VALUES(sno_seq.nextval,'Bond','James','01-Jan-1994’);

- INSERT INTO enrolment
  VALUES(sno_seq.currval,’FIT9132’,...’);
```

Q9. Two new students and their enrolment details need to be added, James Bond wants to enrol in FIT9132 and FIT9001, Bruce Lee only wants to enrol in FIT9132. The sequence for sno is called sno_seq. What problems, if any, exist with this script:

```
-- Add two students
INSERT INTO student VALUES (sno_seq.nextval,'Bond','James','01-Jan-1994');
INSERT INTO student VALUES (sno_seq.nextval,'Lee','Bruce','01-Feb-1994');
-- Add the enrolments
INSERT INTO enrolment VALUES (sno_seq.currval,1,2018,'FIT9132',0,'NA');
INSERT INTO enrolment VALUES (sno_seq.currval,1,2018,'FIT9001',0,'NA');
INSERT INTO enrolment VALUES (sno_seq.currval,1,2018,'FIT9132',0,'NA');
COMMIT;
```

A. There will be an error message. It states that a violation of primary key constraints in the ENROLMENT has occurred.
B. Bruce Lee will be enrolled in FIT9001.
C. There will be NO enrolment record for James Bond.
D. All of the options a-c are problems that will be caused by the script.
E. Some of the options in a-c are problems that will be caused by the script.
F. There will be no problem caused by the script.

PUTTING THIS TO WORK


TEAM
P * team_no NUMERIC (3)
F emp_no NUMERIC (5)
TEAM PK (team_no)
in
has
leads
led_by
EMPLOYEE
P * emp_no NUMERIC (5)
emp_fname VARCHAR (30)
emp_lname VARCHAR (30)
* emp_street VARCHAR (50)
* emp_town VARCHAR (30)
* emp_pcode CHAR (4)
* emp_dob Date
emp_taxno VARCHAR (20)
F team_no NUMERIC (3)
F mentor_no NUMERIC (5)
EMPLOYEE PK (emp_no)
mentored_by
mentors

```
CREATE TABLE employee (
    emp_no       NUMBER(5) NOT NULL,
    emp_fname    VARCHAR2(30),
    emp_lname    VARCHAR2(30),
    emp_street   VARCHAR2(50) NOT NULL,
    emp_town     VARCHAR2(30) NOT NULL,
    emp_pcode    CHAR(4) NOT NULL,
    emp_dob      DATE NOT NULL,
    emp_taxno    VARCHAR2(20),
    team_no      NUMBER(3),
    mentor_no    NUMBER(5)
);

ALTER TABLE employee ADD CONSTRAINT employee_pk PRIMARY KEY ( emp_no );

CREATE TABLE team (
    team_no   NUMBER(3) NOT NULL,
    emp_no    NUMBER(5)
);

ALTER TABLE team ADD CONSTRAINT team_pk PRIMARY KEY ( team_no );
```

```
ALTER TABLE employee
    ADD CONSTRAINT emp_mentors_emp FOREIGN KEY ( mentor_no )
        REFERENCES employee ( emp_no )
            ON DELETE SET NULL;

ALTER TABLE employee
    ADD CONSTRAINT team_has_employee FOREIGN KEY ( team_no )
        REFERENCES team ( team_no )
            ON DELETE SET NULL;

ALTER TABLE team
    ADD CONSTRAINT emp_leads_team FOREIGN KEY ( emp_no )
        REFERENCES employee ( emp_no )
            ON DELETE SET NULL;
```