MONASH University

MONASH
INFORMATION
TECHNOLOGY

Week 6 - Database Implementation (DDL): Creating & Populating the Database

FIT2094 - FIT3171 Databases
Clayton Campus S1 2019.

GROUP OF EIGHT AUSTRALIA

Overview

- **Hour 1**
  - –CLAYTON only: Week 7 arrangements and mid sem break
  - –Basics: CREATE (DDL)
    - •Oracle datatypes
    - •Constraints
    - •Ref Integrity

… then COFFEE BREAK!

- **Hour 2**
  - –ALTER (DDL)
  - –DROP (DDL)
  - –INSERT (DML)
  - –Commit, Rollback
  - –Sequence
  - –Case study

[CLAYTON ONLY] Week 6 chat...

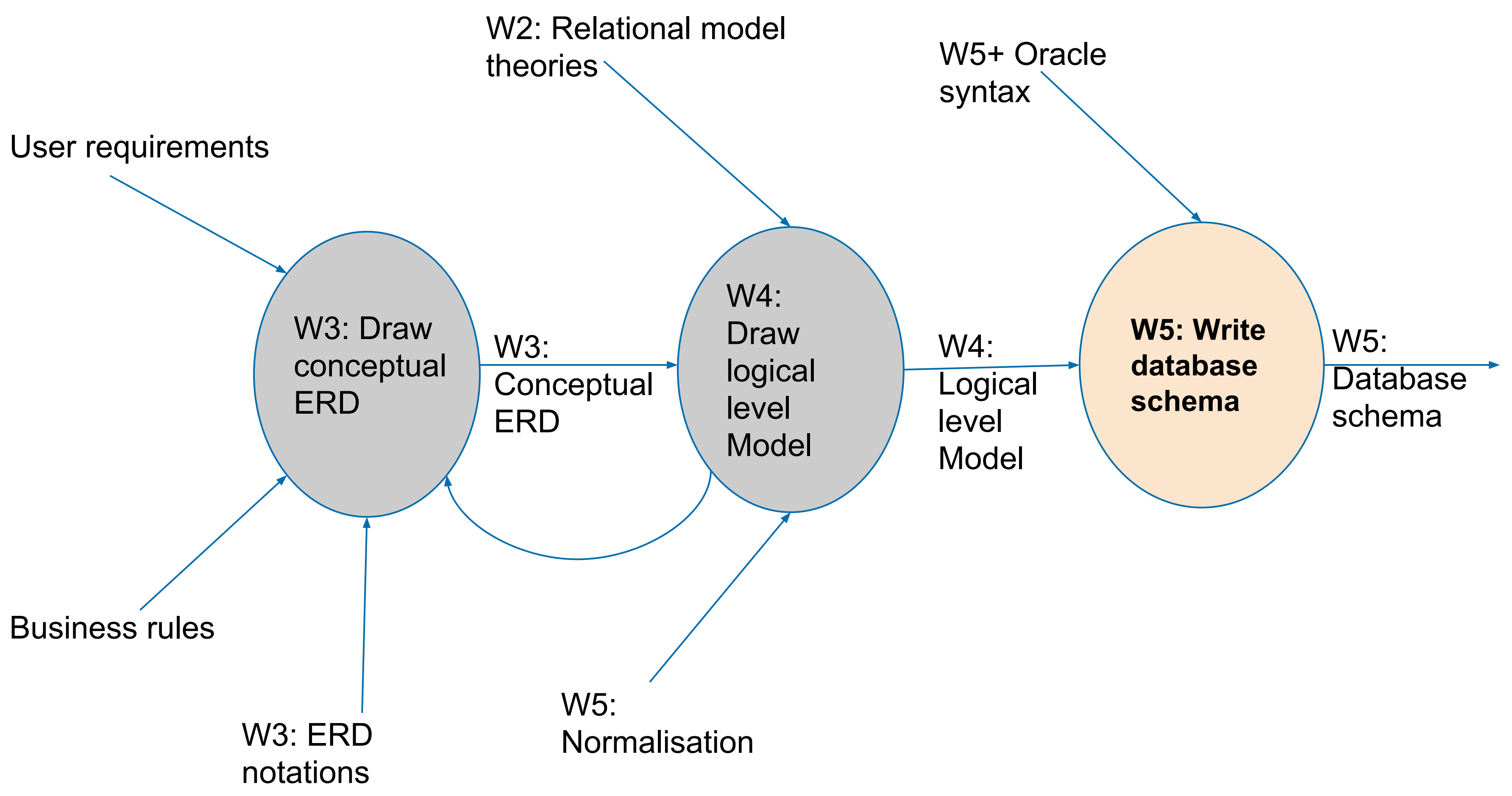
ATTENTION: Week 7 and Good Friday holiday might affect your lecture / tute NEXT week.

- LECTURE: Monday replacement lecture 'workshop' for affected Friday students...
  Other options: livestream / recorded lecture.

- TUTE: More open consultation sessions
  Other options: attend other tutes next week only (subject to seating and other availability constraints)...
  … or do self study at home.

NB: Following week is the MSB :-)

The story so far...



NB: weekly mappings above are approximate.

SQL general syntax

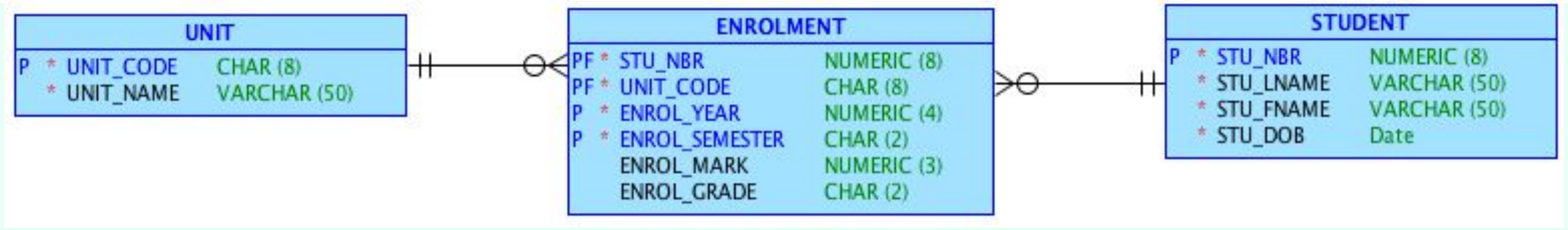
- A single statement is ended with SEMICOLON.
- Predefined KEYWORDs represent clauses (components) of a statement.
- Keywords are NOT case sensitive.
- Examples:

```
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)
  - GRANT

CLAYTON Thursday **6TIBDU** Friday **4AJ0Y6**



Q1. Revision: 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 (DDL)

Img src: @catchstars at Unsplash

```
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)
);
```

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

A. Relation, Attribute, Domain
B. Primary Key
C. Foreign Key
D. Referential Integrity constraint
E. All of the options in A,B,C,D are correct.
F. Some of the options in A,B,C,D are correct.
G. None of the options in A,B,C,D correct!

[Clayton] Audience Q&A

Remember: these datatypes (and some of the SQL statement syntaxes) are ORACLE SPECIFIC

These are transferable skills to other DBMS ‘brands’...
… however code might not work ‘out of the box’ for other ‘brands’.

e.g. Oracle does one-at-a-time with `INSERT INTO` (but supports `INSERT ALL INTO` … for multi insert).

VS

MySQL which does multi-insert with `INSERT INTO`.

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` (literally the whole sentence!)

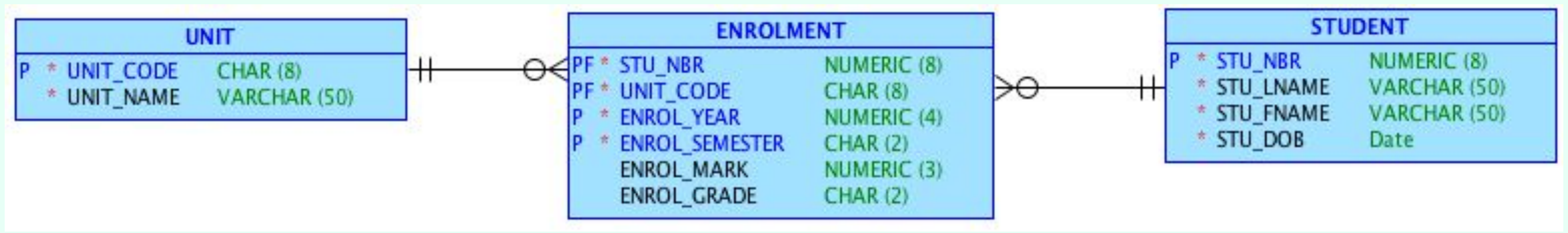
[Clayton Q&A]: **CHAR has unpredictable comparator results due to its "blank-padding" policy. See** https://docs.oracle.com/cd/A97630_01/appdev.920/a96624/b_char.htm

Column Constraints VS Table Level CONSTRAINT

```
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)
);
```

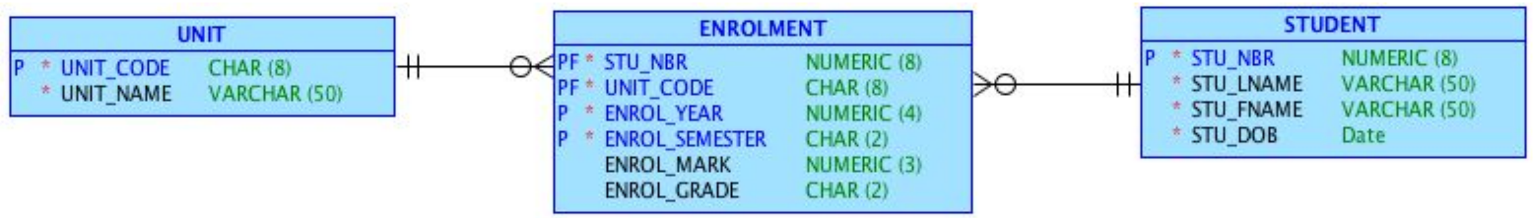
column constraints

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 above diagram? (assuming that we will define the FK as part of the create table statement)

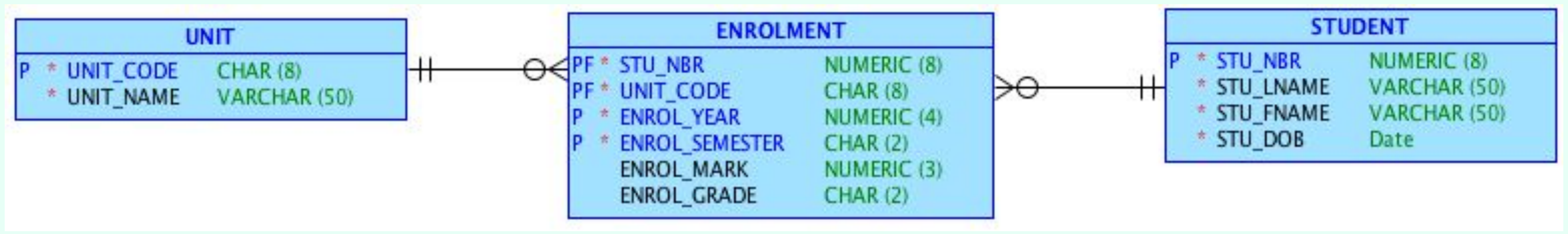
A. UNIT, ENROLMENT, STUDENT
B. ENROLMENT, STUDENT, UNIT
C. STUDENT, UNIT, ENROLMENT
D. UNIT, STUDENT, ENROLMENT
E. More than one option is correct



```
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)
  );
```

== NB: this assumes unit name must be unique too!
== However, consider this - a university may change code but not name...

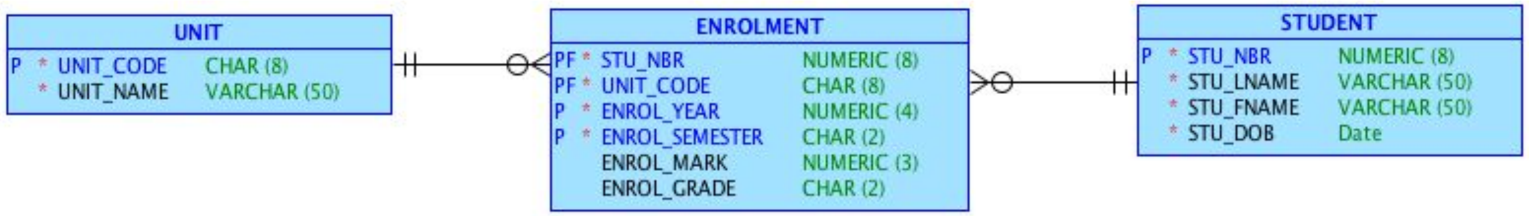
CLAYTON Thursday **6TIBDU** Friday **4AJ0Y6**



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.

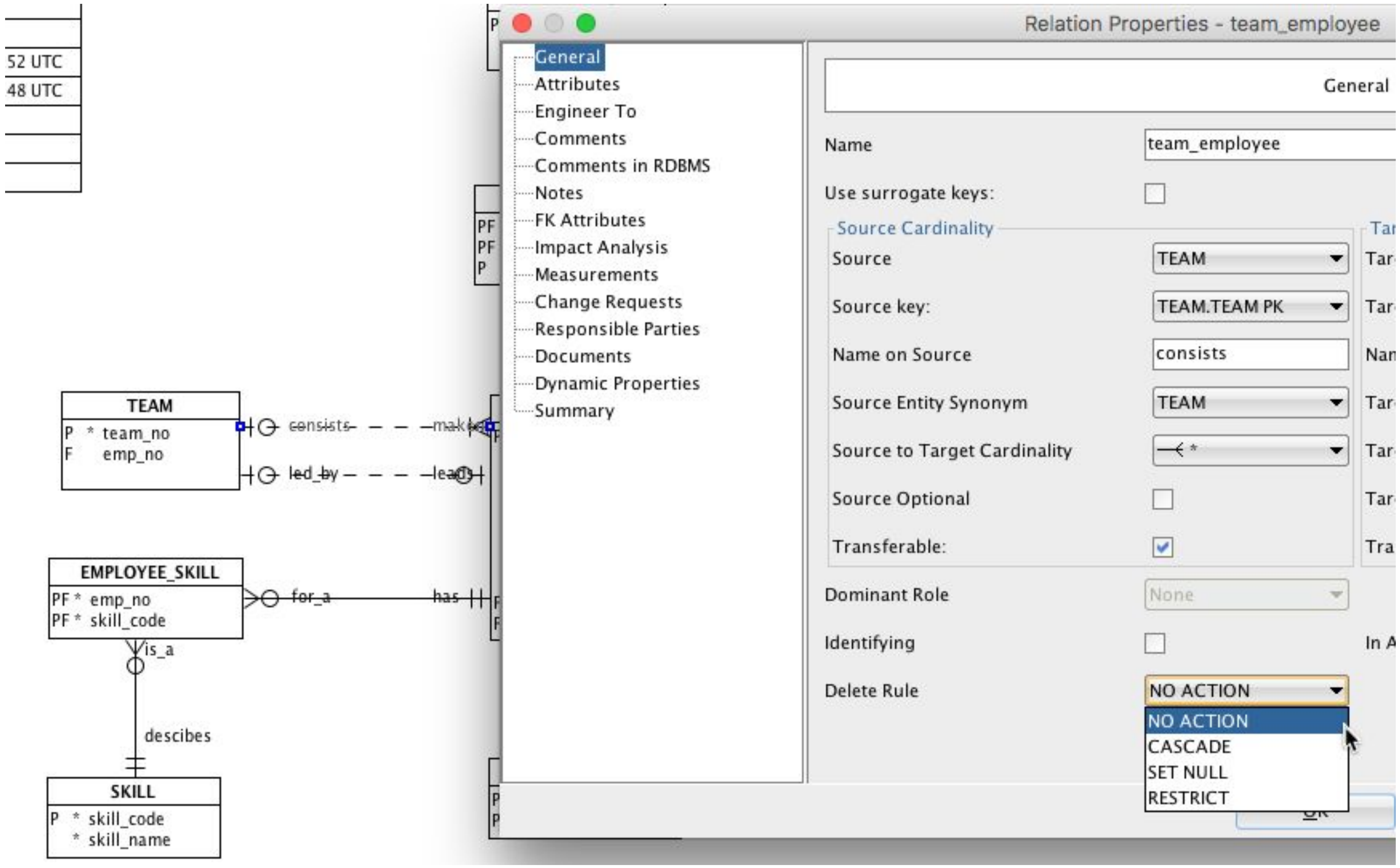
During discussion, name the attribute(s) that will be assigned as FK and what table(s) would it "link"?



```
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)
  );
```

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 e.g. student) if there is corresponding tuple in the table containing the FK (e.g. enrolment).
  - CASCADE
    - A deletion of a tuple in the table referred by the FK (the table containing PK e.g. student) will result in the deletion of the corresponding tuples in the table containing the FK (e.g. enrolment).
  - NULLIFY (Oracle **`SET NULL`**)
    - A deletion of a tuple in the table referred by the FK (the table containing PK e.g. student) will result in the update of the corresponding tuples in the table containing the FK to NULL (e.g. enrolment).



Relation Properties - team_employee

Name	team_employee
Use surrogate keys:	☐
Source	TEAM
Source key:	TEAM.TEAM PK
Name on Source	consists
Source Entity Synonym	TEAM
Source to Target Cardinality	—< *
Source Optional	☐
Transferable:	✓
Dominant Role	None
Identifying	☐
Delete Rule	NO ACTION

[Clayton] Audience Q&A

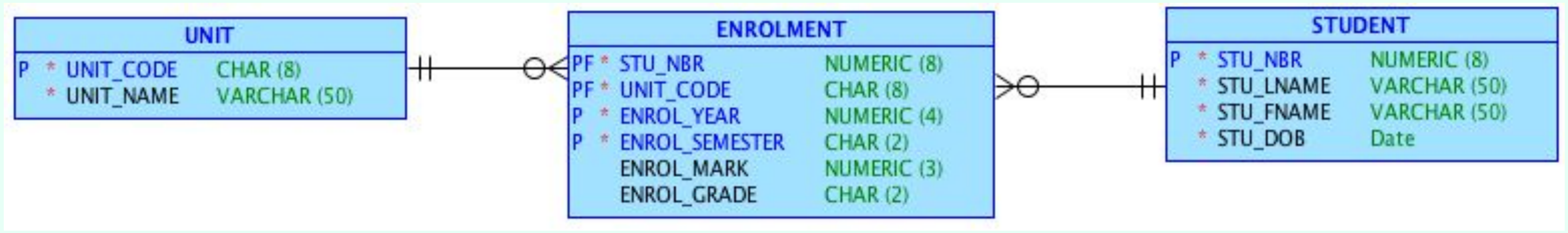
Using SQLFiddle (online editor), Oracle 11g

Demo on
`NO ACTION` - error (why?)
`CASCADE` - ok
`SET NULL` - ok (discuss!)

IMPORTANT: Oracle != other brands in terms of behaviour… MySQL supports other keywords e.g. `RESTRICT`

```
-- DDL: very simple table for clarity! Note that surrogate keys are used as demo;
-- else "SET NULL" will fail by definition as compound PKs can't have a NULL component!!!

CREATE TABLE student
  (
    stu_nbr      NUMBER(8)        NOT NULL,
    stu_name     VARCHAR(50)      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),
    unit_code            CHAR(8) NOT NULL,
    surrogate_key        CHAR(11),
    CONSTRAINT pk_enrolment PRIMARY KEY (surrogate_key),
    CONSTRAINT fk_enrolment_student FOREIGN KEY (stu_nbr)
        REFERENCES student (stu_nbr) ON DELETE SET NULL,
    CONSTRAINT fk_enrolment_unit    FOREIGN KEY (unit_code)
        REFERENCES unit (unit_code) ON DELETE SET NULL
  );


-- DML
INSERT INTO student VALUES (10000001, 'Brendon');
INSERT INTO student VALUES (10000002, 'Donald');
INSERT INTO student VALUES (10000003, 'Kanye');

INSERT INTO unit VALUES ('FIT1001', 'Intro to Programming');
INSERT INTO unit VALUES ('ECO1001', 'Intro to Economics');
INSERT INTO unit VALUES ('MUS1001', 'Intro to Hip Hop');

INSERT INTO enrolment VALUES (10000001, 'FIT1001', 'ENROL201901');
INSERT INTO enrolment VALUES (10000002, 'ECO1001', 'ENROL201915');

DELETE FROM student WHERE stu_nbr=10000001;

-- should affect student, then enrolment, depending on DELETE rule
SELECT * from student;
SELECT * from enrolment;
```

CLAYTON Thursday **6TIBDU** Friday **4AJ0Y6**



Q5. Assume that the table ENROLMENT contains enrolment details for students in FIT3171 and FIT2001. The referential integrity constraint is `CASCADE`. What would happen to tuples in ENROLMENT with the unit_code='FIT3171' when we delete the FIT3171 record from UNIT?

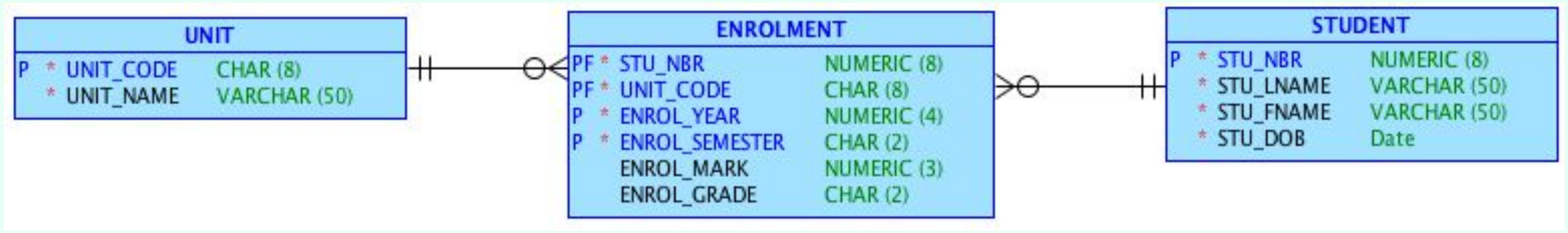
A. They will be deleted.

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.

MONASH University



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.

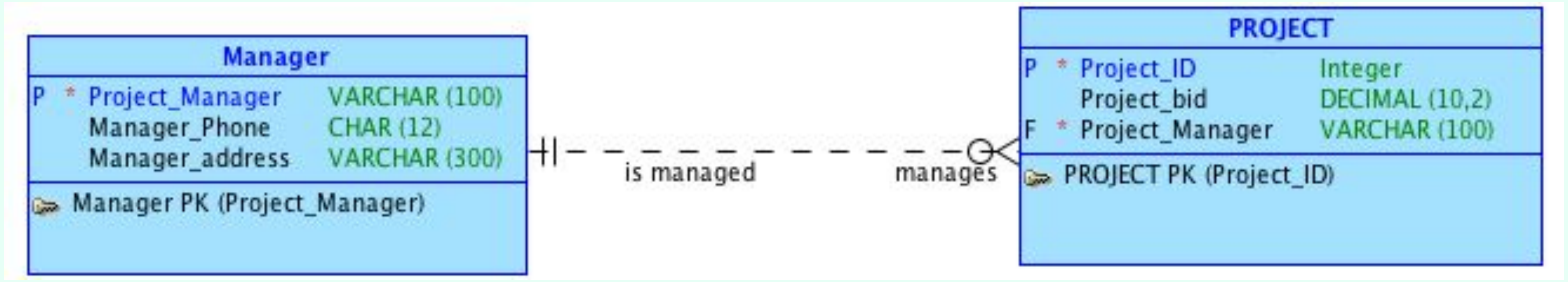
C. The stu_nbr='1234' in the STUDENT table will be updated to NULL.

D. Deletion will not be permitted by the DBMS.

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.

CLAYTON Thursday **6TIBDU** Friday **4AJ0Y6**


Manager
P * Project_Manager VARCHAR (100)
Manager_Phone CHAR (12)
Manager_address VARCHAR (300)
Manager PK (Project_Manager)
is managed
manages
PROJECT
P * Project_ID Integer
Project_bid DECIMAL (10,2)
F * Project_Manager VARCHAR (100)
PROJECT PK (Project_ID)

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?
(Hint: the fine print should give you another hint… :-)

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

MONASH University

Coffee break - see you in 10 minutes.

Img src: Jazmin Quaynor @jazminantoinette at Unsplash

ALTER (DDL)

Img src: @katyukawa at Unsplash

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'))
            );
```

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)
    );
```

ALTER for 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
    );
```

DROP (DDL)

Img src: The Guardian / C-SPAN

DELETING A TABLE

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

INSERT ... (DML)
Adding tuples (‘rows’) to table.

Img src: @samuelzeller at Unsplash / icons8

INSERT

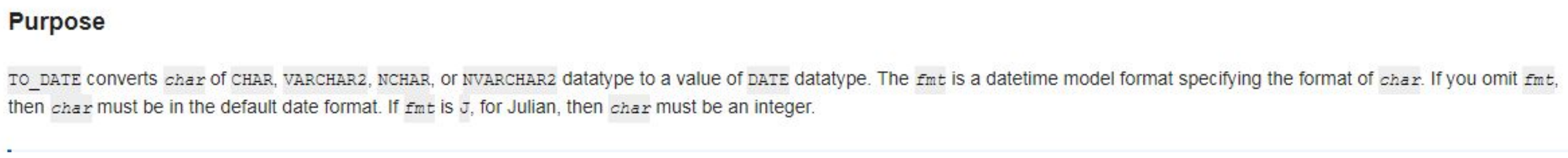
- Adding data to a table in a database.
- SYNTAX:

```
INSERT INTO table [(column [, column...])]
      VALUES (value [, value...]);
```

```
INSERT INTO unit VALUES ('FIT3171', 'Databases');
INSERT INTO student VALUES (112233, 'Wild',
          'Wilbur','01-Jan-1995')
```

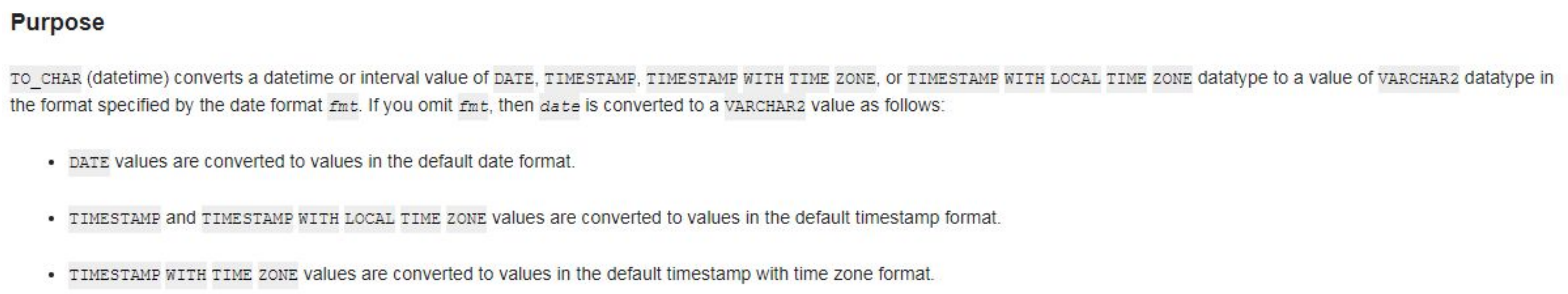
What about consistency of dates? Say we used ‘`01-01-1995`’?
Role of: to_date and to_char

INSERT - helper functions for dates



Purpose

`TO_DATE` converts *`char`* of `CHAR`, `VARCHAR2`, `NCHAR`, or `NVARCHAR2` datatype to a value of `DATE` datatype. The *`fmt`* is a datetime model format specifying the format of *`char`*. If you omit *`fmt`*, then *`char`* must be in the default date format. If *`fmt`* is `J`, for Julian, then *`char`* must be an integer.



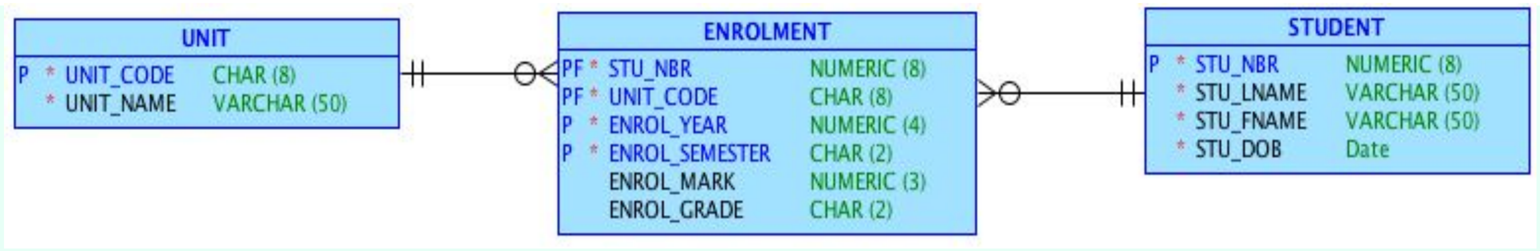
Purpose

`TO_CHAR` (datetime) converts a datetime or interval value of `DATE`, `TIMESTAMP`, `TIMESTAMP WITH TIME ZONE`, or `TIMESTAMP WITH LOCAL TIME ZONE` datatype to a value of `VARCHAR2` datatype in the format specified by the date format *`fmt`*. If you omit *`fmt`*, then *`date`* is converted to a `VARCHAR2` value as follows:

- `DATE` values are converted to values in the default date format.
- `TIMESTAMP` and `TIMESTAMP WITH LOCAL TIME ZONE` values are converted to values in the default timestamp format.
- `TIMESTAMP WITH TIME ZONE` values are converted to values in the default timestamp with time zone format.

https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions183.htm
https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions180.htm

CLAYTON Thursday **6TIBDU** Friday **4AJ0Y6**



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,
                'FIT3171',1,2018,45,'N');
INSERT INTO enrolment VALUES (112233,
                'FIT2001',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,'FIT3171',...');
```

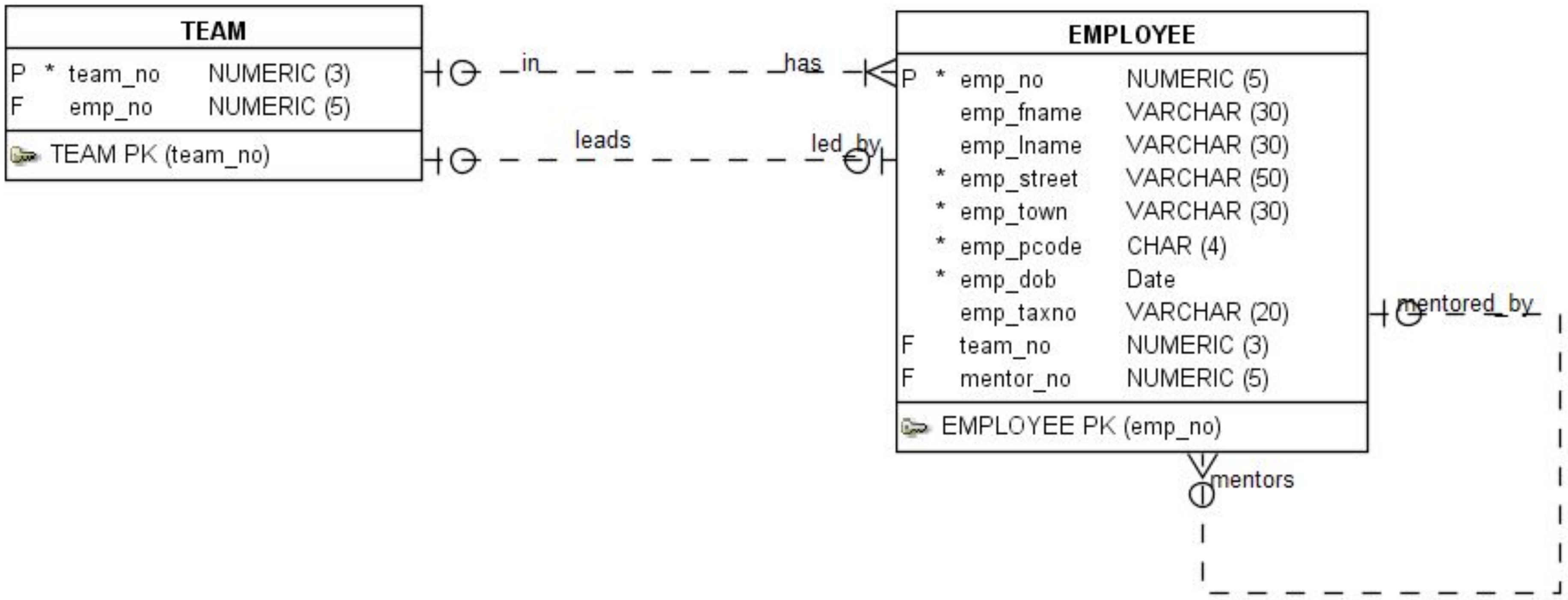
Q9. Two new students and their enrolment details need to be added, James Bond wants to enrol in FIT3171 and FIT2001, Bruce Lee only wants to enrol in FIT3171. The sequence for sno is called sno_seq. What problems, if any, exist with this script:

```
-- 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,’FIT3171’,0,’NA’);
INSERT INTO enrolment VALUES (sno_seq.currval,1,2018,’FIT2001’,0,’NA’);
INSERT INTO enrolment VALUES (sno_seq.currval,1,2018,’FIT3171’,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 FIT2001.
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.

Case Study: DDL of our Monash Software Case (since Week 3)

Img src: @samuelzeller at Unsplash



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;
```