FIT2094-FIT3171 2019 S1 -- Week 6 eBook

Credits and Copyrights:
Authors: FIT3171 2018 S2 UG Databases
FIT Database Teaching Team
Maria Indrawan, Manoj Kathpalia, Lindsay Smith, et al

Change Log:

- Formatted and Imported from Alexandria March 2019.

6.0. Normalisation Tutorial Activities: Introduction

Credits for the Updated Case Study version courtesy of Lindsay Smith and Team.

Normalisation process starts with identifying attributes and the possible existence of the repeating group. The starting set of attributes is called UNF (un-normalised form). During analysis, it is possible to have different sets of un-normalised form (UNF).

For example, consider the following project management form.

PROJECT_CODE	PROJECT_MANAGER	MANAGER_PHONE	MANAGER_ADDRESS	PROJECT_BID_PRICE
21-5Z	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$16,833,460.00
25-2D	Jane D. Grant	615-898-9909	218 Clark Blvd., Nashville, TN 36362	$12,500,000.00
25-5A	George F. Dorts	615-227-1245	124 River Dr., Franklin, TN 29185	$32,512,420.00
25-9T	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$21,563,234.00
27-4Q	George F. Dorts	615-227-1245	124 River Dr., Franklin, TN 29185	$10,314,545.00
29-2D	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$25,559,999.00
31-7P	William K. Moor	904-445-2719	216 Morton Rd., Stetson, FL 30155	$56,850,000.00

This form shows multiple instances of the details about PROJECTs, as such the UNF is represented as:

PROJECT(project_code, project_manager, manager_phone, manager_address, project_bid_price)

To complete the normalisation, you take the UNF as the starting point and follow the normalisation steps through to at least 3NF.

6.0.1. The Normalisation process:

Legend: Primary Key is underlined, *Foreign Key* is in italic.
The identification of the Foreign Key is *not part of the formal steps of normalisation*, however, it will give you a mechanism to check whether you have split the attributes of a relation into two separate relations correctly. When you split a relation into two, there

should be at a common attribute (or attributes) in both relations. In one relation, it will be a Primary Key (PK). In the other relation, the attribute will be a Foreign Key (FK).

When completing normalisation you **must not add any new attributes**, for example, surrogate keys (these are added to the final logical model post normalisation). Also during normalisation you **must not exclude any attributes** displayed on the provided form/s - we wish to capture the full semantics of the form/s. Decisions about making an attribute derived or not need to be based on transaction loads and more advanced statistics which are not available at this stage of the design process. Such decisions are made after the logical model is implemented as part of the fine-tuning of the database.

You should complete your normalisations using either Google Docs or MS Word.

6.0.2. UNF

PROJECT(project_code, project_manager, manager_phone, manager_address, project_bid_price)

6.0.3. 1NF

PROJECT(<u>project_code</u>, project_manager, manager_phone, manager_address, project_bid_price)

Dependency diagrams: project_code → project_bid_price, project_manager FULL

project_manager → manager_phone, manager_address TRANSITIVE

6.0.4. 2NF

There is no partial dependency, the 2NF is the same as the 1NF. Note that you are **required** to show all forms, even if they are the same as a previous form.

PROJECT(project_code, project_manager, manager_phone, manager_address, project_bid_price)

6.0.5. 3NF

PROJECT(project_code, project_bid_price, *project_manager*)

MANAGER(project_manager**,** manager_phone, manager_address)

6.2. Tutorial Questions.

6.2.1. This exercise is adapted from Connolly and Begg, Exercise 14.15 (p441)

staffNo	dentistName	patNo	patName	appointment		surgeryNo
				date	time	
S1011	Tony Smith	P100	Gillian White	12-Sep-08	10:00	S15
S1011	Tony Smith	P105	Jill Bell	12-Sep-08	12:00	S15
S1024	Helen Pearson	P108	Ian MacKay	12-Sep-08	10:00	S10
S1024	Helen Pearson	P108	Ian MacKay	14-Sep-08	14:00	S10
S1032	Robin Plevin	P105	Jill Bell	14-Sep-08	16:30	S15
S1032	Robin Plevin	P110	John Walker	15-Sep-08	18:00	S13

1. Describe possible insertion, deletion and update anomalies for the DENTIST relation.
2. Write the UNF (un-normalised Form) for this form. Then perform the normalisation process up to the 3NF.
3. Draw the dependency diagram of the relation(s) at 1NF. Since you do not have access to further data about this scenario, your answer needs to be based on the data supplied.
4. Draw all of the 3NF relations you have found as a logical level diagram with crows-foot notation. Identify clearly the Primary Keys and the Foreign Keys in the diagram.

6.2.2. University Database Exercise

1. Normalise the following forms describing a student/units system to 3NF. Remember you must show UNF, 1NF, 2NF, 3NF and supply the dependency diagrams at 1NF for each form.

UNITS CURRENTLY APPROVED — **REPORT DATE:** 9/07/2015

Unit Number	Unit Name	Unit Description	Unit Value
FIT9131	Programming Foundations	Introduction to programming	6
FIT9132	Introduction to Databases	Database Fundamentals	6
FIT9134	Computer Architecture and Operating Systems	Fundamentals of computer systems and the computing environment	6
FIT9135	Data Communications	Fundamentals of data and computer communications	6

* Unit values may be either 3, 6 or 12 points

LECTURER DETAILS **REPORT DATE:** 29/07/2015

LECTURER'S NUMBER: 10234

LECTURER'S NAME: GUISEPPE BLOGGS

LECTURER'S OFFICE No.: 169

LECTURER'S PHONE No.: 99037111

UNIT ADVISER FOR:

UNIT NUMBER	UNIT NAME
FIT9131	Programming Foundations
FIT9134	Computer Architecture and Operating Systems

* A given unit may have several advisers

* Some lecturers share offices, although each have their own phone

STUDENT DETAILS **REPORT DATE:** 29/07/2015

STUDENT No.: 12345678

STUDENT NAME: Poindexter Jones

STUDENT ADDRESS: 23 Wide Road, Caulfield, 3162

COURSE ENROLLED: MIT

MODE OF STUDY: On-Campus

MENTOR NUMBER: 10234

MENTOR NAME: Guiseppe Bloggs

ACADEMIC RECORD:

UNIT NUMBER	UNIT NAME	YEAR / SEMESTER	GRADE
FIT9131	Programming Foundations	2014/2	N
FIT9131	Programming Foundations	2015/1	D
FIT9132	Introduction to Databases	2015/1	D

* Grade may have the value N, P, C, D or HD

* Mode of Study must be On-campus (O) or Distance Education (D)

In order to add a student, the lecturer who advises (supervises or teaches) that student must already exist in the database. No lecturer may be deleted who advises any students which are currently in the database. If the lecturer number of a lecturer is changed, then the number would be changed for each student advised by that lecturer.

Note:

- To simplify the normalisation process that involves multiple forms, you should perform the normalisation one form at a time until all relations are in 3NF. Once

you have done this process for all forms, consolidate the relations from the different forms by:

    - group together all relations with the same primary key, i.e representing the same entity.
- choose a single name for synonyms. For example, **mentor** is the same as **lecturer.**

After preparing a conceptual model the next stage is to select the type of database we will use (for us: relational) and convert our conceptual model into an appropriate logical model. For logical modelling, we will make use of Oracle SQL Developer Data Modeler.

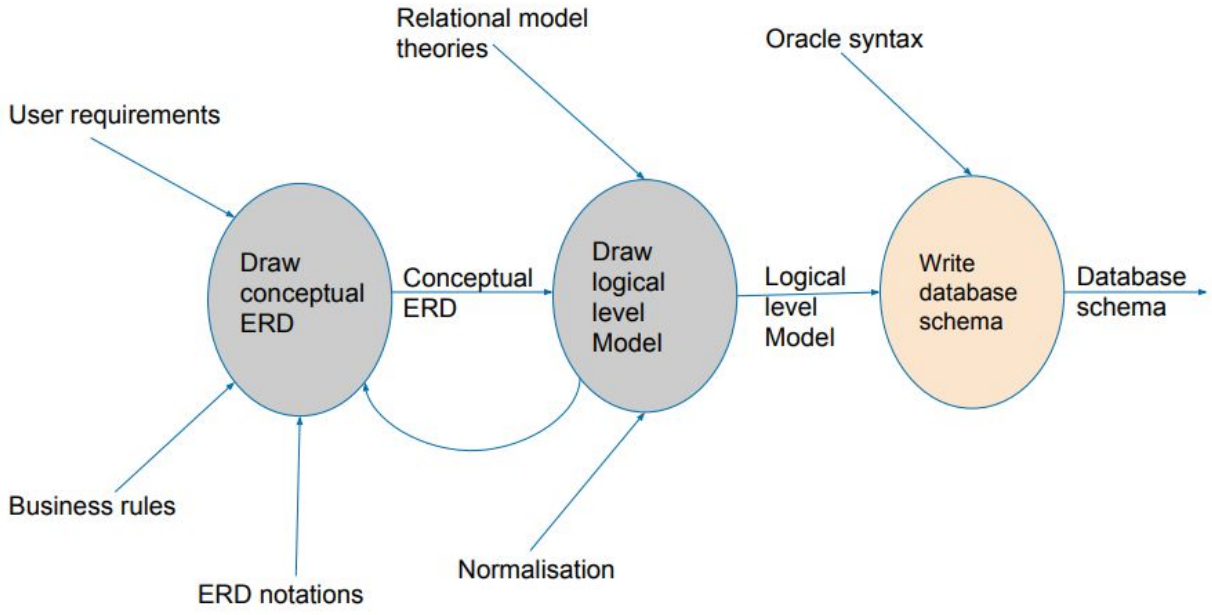
Full credits for the detailed writeup and screenshots to Lindsay Smith and team.

6.3. Pre-Lecture Notes

These are notes you may find useful to read through before the lecture, adapted from Lindsay Smith's lecture material. NOTE: THESE ARE NOT THE FINAL LECTURE SLIDES.

Read up the pre-lecture notes below.

IMPORTANT: AS THE LECTURE FOCUSES ON THE 'FLIPPED CLASSROOM' APPROACH - I.E. MORE INTERACTIVE DISCUSSION AND LESS ON DISCOVERING NEW MATERIAL - THE THEORY SLIDES BELOW ARE PROVIDED FOR YOUR READING CONVENIENCE BEFORE THE LECTURE.



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

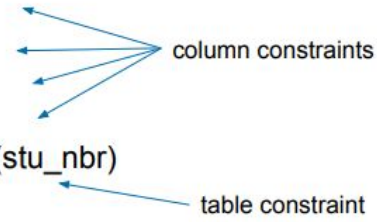
- Data Definition Language (DDL)
  - Creating database structure.
    - CREATE TABLE, ALTER TABLE, DROP TABLE
- Data Manipulation Language (DML)
  - Adding and Manipulating database contents (rows).
    - INSERT, UPDATE, DELETE
  - Retrieving data from database
    - SELECT
- Data Control Language (DCL)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

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

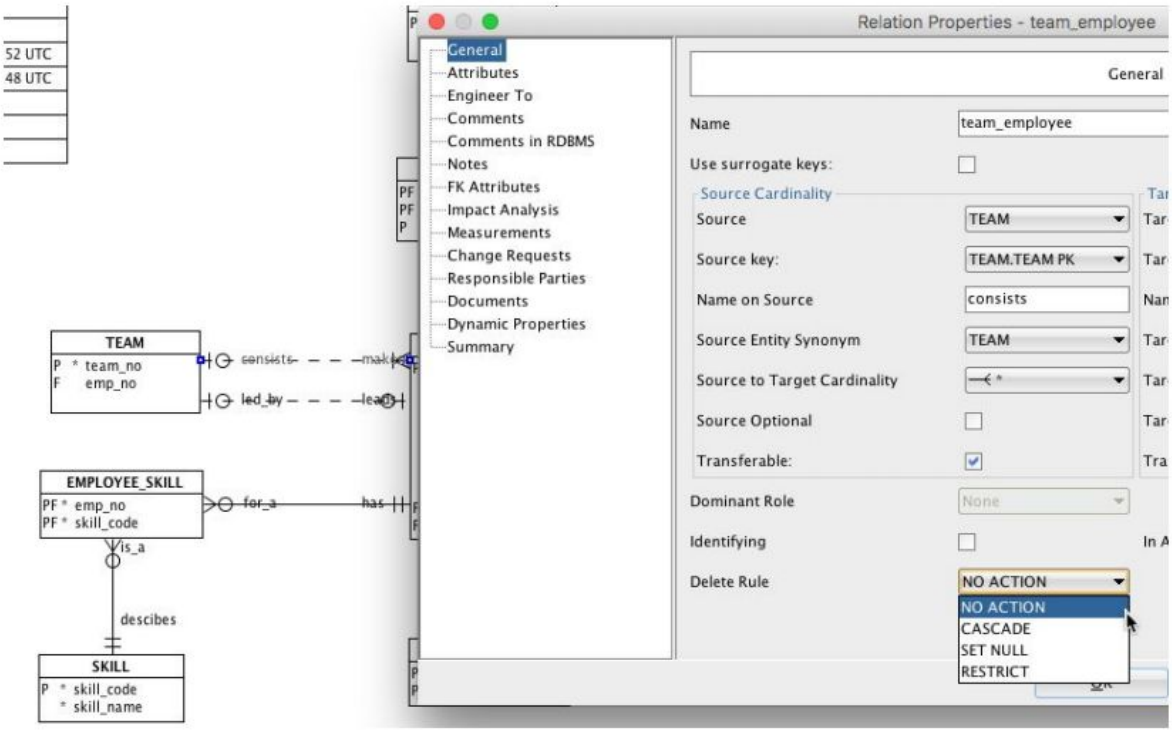
Column VS Table Level Constraints

```
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)
);
```



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.



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.

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;`

INSERT

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

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

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

```
INSERT INTO unit VALUES ('FIT2094', 'Databases');
INSERT INTO student VALUES (112233, 'Wild',
          'Wilbur','01-Jan-1995')
```

Role of: to_date and to_char

COMMIT and ROLLBACK

```
INSERT INTO enrolment VALUES (112233,
                'FIT2094',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

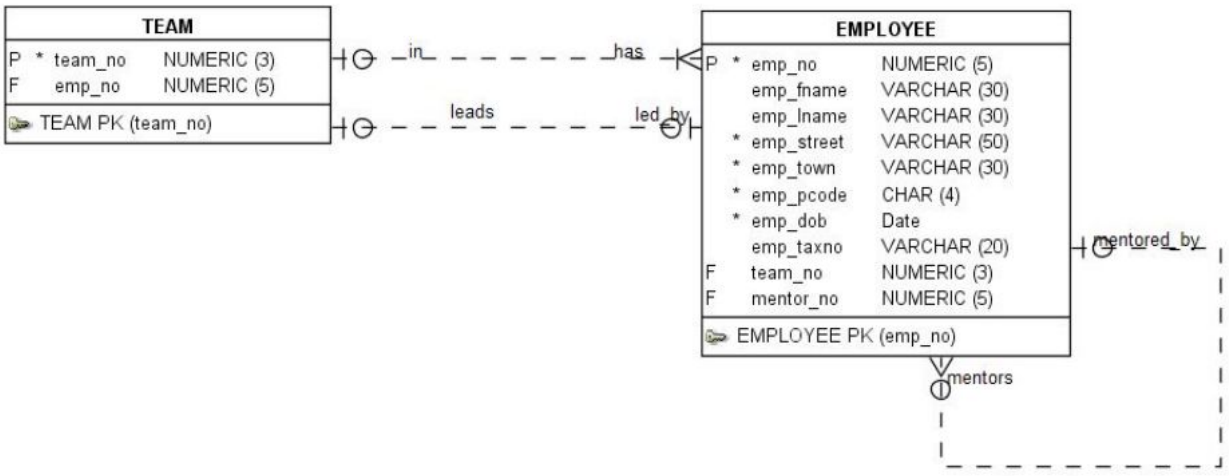
```
CREATE SEQUENCE sno_seq
INCREMENT BY 1;
```

  - 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,'FIT2094',...');
```



EOF.