MONASH University

MONASH
INFORMATION
TECHNOLOGY

# Creating & Populating the Database – Data Definition Language

Lindsay Smith

GROUP
OF EIGHT
AUSTRALIA

# 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 Manipulation Language (DML)
    - GRANT

# CREATE A TABLE (DDL)

# Common ORACLE data types

- **Text:** CHAR(size), VARCHAR2(size)
  - – e.g., CHAR(10), VARCHAR2(10)
  - – CHAR(10) → 'apple' = 'apple     '
  - – VARCHAR2(10) → 'apple' != 'apple     '
- **Numbers:** NUMERIC, NUMBER(precision, scale)
  - – Weight NUMBER → Weight = 7456123.89
  - – Weight NUMBER(9) → Weight = 7456124
  - – Weight NUMBER(9,2) → Weight = 7456123.89
  - – Weight NUMBER(9,1) → Weight = 7456123.9
- **Data/Time:** DATE, TIMESTAMP
  - – DATE can store a date and time
  - – 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

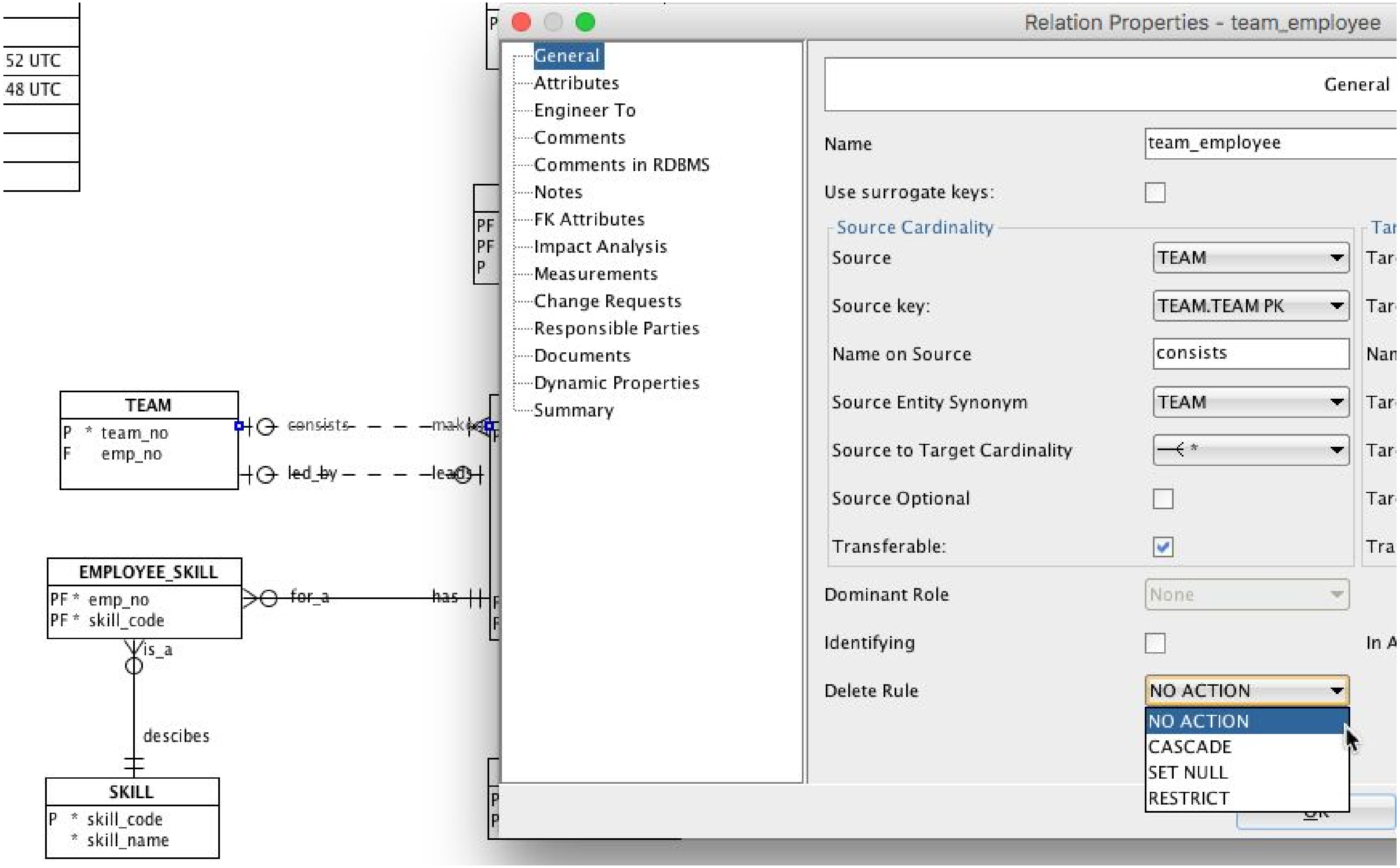
# Alternative method of defining FKs

```
CREATE
 TABLE enrolment
 (
  stu_nbr             NUMBER(8)    NOT NULL,
  unit_code           CHAR(7)      NOT NULL,
  enrol_year                NUMBER(4)    NOT NULL,
  enrol_semester            CHAR(1)      NOT NULL,
  mark                NUMBER(3),
  grade               CHAR(3),
  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.



Relation Properties - team_employee

General

Name: team_employee

Use surrogate keys: ☐

Source Cardinality

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

- NO ACTION
- CASCADE
- SET NULL
- RESTRICT

# 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

- It is 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 ('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, ‘FIT1004’,1,2012,45,’N’);
INSERT INTO enrolment VALUES (112233, ‘FIT1001’,1,2012,80,’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,’FIT1004’,...’);
        ```

# MODIFYING TUPLES USING UPDATE AND DELETE

# UPDATE

- Changes the value of existing data.
- For example, at the end of semester, change the mark and grade from null to the actual mark and grade.

```
UPDATE table
SET column = (subquery) [, column = value, ...]
[WHERE condition];
```

```
UPDATE enrolment
SET mark = 80,
      grade ='HD'
WHERE sno = 112233
  and ……
```

```
UPDATE enrolment
SET mark = 85
WHERE unit_code = (SELECT unit_code FROM unit WHERE
          unit_name='Introduction to databases')
           AND mark = 80;
```

# DELETE

- Removing data from the database

DELETE FROM *table*
[WHERE *condition];*

```
DELETE FROM enrolment
WHERE sno='112233'
         AND
              unit_code= (SELECT unit_code
         FROM unit
         WHERE unit_name='Introduction to Database' )
         AND
              semester='1'
         AND
               year='2012';
```