COS221
L09 - Introduction to SQL
(Chapter 4 in Edition 6 and Chapter 6 in Edition 7)

Linda Marshall

16 March 2023

SQL Background

- SQL (or Structured Query Language which is a shortened form of the original name SEQUEL (Structured English QUEry Language)) forms the backbone of Relational Database Systems.
- There are differences in implementations of SQL in available RDBMS's. The fundamentals, however, are the same and most SQL programs are easily transferred between RDBMS's.
- SQL provides a high-level declarative language stating what the query is and not how it must happen.
- SQL is a comprehensive database language. It can be used for data definitions, queries and updates. It is therefore a DDL and a DML. Additionally, views can be defined.
- Over the years, SQL has been extended to provide functionality for data warehousing, data mining, multimedia etc.

Data definitions and data type in SQL - Terminology

SQL uses the terminology of:

- Table (previously seen as Relation in the Relational Model and Entity type in the (E)ER model
- Row (referred to as a Tuple in the Relational Model and part of the Entity set in (E)ER)
- Column (thankfully Attributes in both the Relational Model and (E)ER)

The CREATE statement is used to create schemas, tables, types and domains. Additionally the CREATE statement is used for constructs such as views, assertions and triggers.

Schemas and Catalogues

A *schema* is used to group together tables and other constructs that fall under the same database application

- An SQL schema is identified by a name, an authorisation identifier - the user who owns the schema, and descriptors for each element in the schema.
- Schema elements include tables, types, constraints, views, domains, and other constructs - such as authorisation - to describe the schema.

A *catalogue* is a named collection of schemas

- Most users are moved directly into a schema they work with when they log into the database system
- Contains a schema INFORMATION_SCHEMA. This schema provides information on all the schemas in the catalog and all the element descriptors in these schemas.

Note: Integrity constraints such as referential integrity can be defined between relations only if they exist in schemas within the same catalog. Schemas within the same catalog can also share certain elements, such as type and domain definitions.

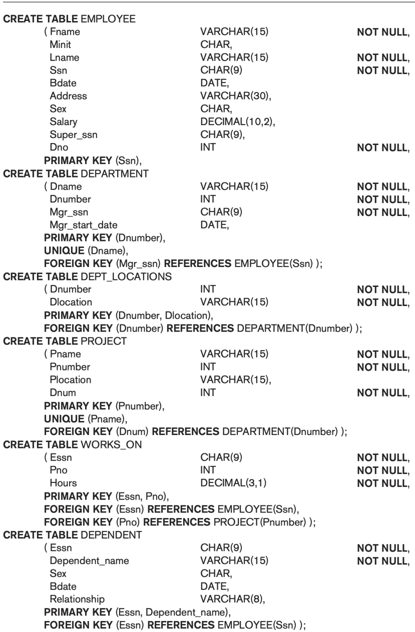
Creating a Schema

- Schemas are created with the CREATE SCHEMA statement.
- This statement may include all the elements in the schema or may include only the schema name and authorisation identifier CREATE SCHEMA COMPANY AUTHORIZATION 'John.Smith';
- Typically not all users are authorised to create schemas. Permissions to do so are assigned to a user by the DBA.
- Once a schema has been created, tables can be added to the schema.

Creating a Table

- The CREATE TABLE command is used to specify a new table/relation. Each new table is given a name, attributes and constraints.
- Attributes are specified first and are given a name, data type to specify the domain of values and possibly attribute constraints, e.g. NOT NULL.
- Constraints - key, entity integrity and referential integrity - are specified after the attributes are declared. Constraints can also be added later with the ALTER TABLE command.
- The schema name can be included when creating a table. If it is left out, the current schema is used, for example:
  CREATE TABLE COMPANY.EMPLOYEE
  or
  CREATE TABLE EMPLOYEE

Creating a Table



```
CREATE TABLE EMPLOYEE
        ( Fname                 VARCHAR(15)         NOT NULL,
          Minit                 CHAR,
          Lname                 VARCHAR(15)         NOT NULL,
          Ssn                   CHAR(9)             NOT NULL,
          Bdate                 DATE,
          Address               VARCHAR(30),
          Sex                   CHAR,
          Salary                DECIMAL(10,2),
          Super_ssn             CHAR(9),
          Dno                   INT                 NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)         NOT NULL,
          Dnumber               INT                 NOT NULL,
          Mgr_ssn               CHAR(9)             NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                 NOT NULL,
          Dlocation             VARCHAR(15)         NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)         NOT NULL,
          Pnumber               INT                 NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                 NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)             NOT NULL,
          Pno                   INT                 NOT NULL,
          Hours                 DECIMAL(3,1)        NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)             NOT NULL,
          Dependent_name        VARCHAR(15)         NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Figure 6.1
SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 5.7.

Creating a Table

- Tables created using CREATE TABLE are referred to as base tables (or base relations) as opposed to virtual tables/relations created using the CREATE VIEW command.

Notes:

- Attributes in a base relation are ordered in the sequence they are specified.
- Some attributes may cause errors when created due to referring to tables that have not been created yet. In such cases, defining the attributes and then adding the constraints with ALTER TABLE solves this problem. Examples of this problem are:
  - creating a foreign key for Super_ssn before the table is created - circular reference; or
  - creating the foreign key for department number in EMPLOYEE before the DEPARTMENT table is created

Basic Datatypes

Basic datatypes include:

- Numeric
- Character string
- Bit string
- Boolean
- Date

Basic Datatypes

- Numeric: INTEGER (or INT), SMALLINT, FLOAT (or REAL), DECIMAL(i,j) (or DEC(i,j)), NUMERIC(i,j) where *i* is the precision (total number of decimal digits) and *j* is the scale (number of digits after the decimal point, default is 0)
- Character string: Fixed length - CHAR(n) (or CHARACTER(n), Varying length - VARCHAR(n) (or CHAR VARYING(n)), CLOB(nS), where S is specified in kilobytes (K), ... , gigabytes (G). CHAR's are generally padded with blanks which can be are ignored. A concatenation operator (||) exists to join two 'strings'
- Bit string: These can either be a fixed size (BIT(n)) or varying (BIT VARYING(n)). Literal bit strings are prefixed with B' (B followed by a single quotation mark) and suffixed with a ' (a single quotation mark). BLOBs, or BINARY LARGE OBJECTS are used to for a bitstream as images. Again in K, ..., G.

Basic Datatypes

- Boolean: BOOLEAN can be TRUE, FALSE or UNKNOWN (referred to as three value logic). The latter is necessary because of the presence of NULL values.
- Date: DATE has the form YYYY-MM-DD. Additionally there is a TIME (HH:MM:SS or TIME(i) where i is given in seconds) type which could be qualified with WITH TIME ZONE. If no time zone is specified the SQL session time zone is used. Values of date types can be compared ($<$).

Other data types

- Other data types, which may be system specific, may include:
  - Timestamp
  - Interval - relative value that is used to increment or decrement an absolute value.
- Other than specifying the data type directly, domains can be defined. This improves readability. For example: CREATE DOMAIN SSN_TYPE AS CHAR(9);
- It is also possible to create user defined types with the CREATE TYPE command

Specifying constraints in SQL

Basic constraints are checked using the CHECK clause at table creation, these include:

- key and referential integrity,
- restrictions on attribute domains and NULL's
- constraints on individual tuples within a relation.

Constraints can be given a name by using the CONSTRAINT keyword followed by the constraint name. Naming constraints is optional.

Specifying constraints - Key and Referential Integrity

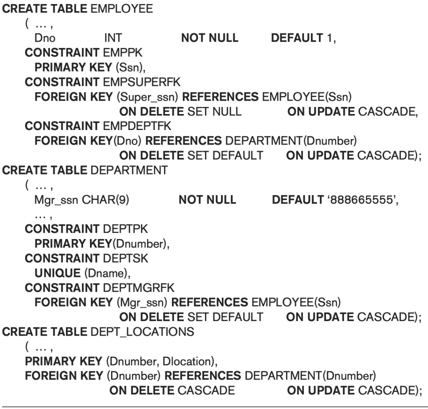
Makes use of the PRIMARY KEY clause to specify if an attribute is a primary key (or an attribute of a multi-attribute primary key).

- To specify Dnumber as the primary key of DEPARTMENT, the following is used:
  Dnumber INT PRIMARY KEY,
- To specify alternate keys (candidate keys), the UNIQUE clause is used, for example:
  Dname VARCHAR(15) UNIQUE,
- Referential integrity is specified using the FOREIGN KEY clause. The default action when an integrity violation occurs is reject the update, this is known as the RESTRICT option. The schema designer may specify alternative actions by attaching a referential integrity action to the foreign key constraint. The options include: SET NULL, CASCADE, and SET DEFAULT. The option must be qualified with ON DELETE or ON UPDATE.

Specifying constraints - Attribute constraints and defaults

- If NULL is not permitted for a specific attribute a constraint NOT NULL may be specified. NOT NULL is always specified for attributes of primary keys.
- Default values are specified by appending the DEFAULT <value> clause to an attribute definition. Default values are used if an explicit value is not provided for that attribute.
  Dnumber INT NOT NULL CHECK (Dnumber > 0 AND Dnumber < 21);
  or create a domain and use it as the attribute type :
  CREATE DOMAIN D_NUM AS INTEGER CHECK (D_NUM > 0 AND D_NUM < 21);

Specifying constraints - Attribute constraints and defaults



```
CREATE TABLE EMPLOYEE
    ( … ,
      Dno          INT          NOT NULL      DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                      ON DELETE SET NULL          ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                      ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( … ,
      Mgr_ssn CHAR(9)           NOT NULL      DEFAULT '888665555',
      … ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
      FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
                      ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
    ( … ,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
                    ON DELETE CASCADE             ON UPDATE CASCADE);
```

Figure 6.2
Example illustrating how default attribute values and referential integrity triggered actions are specified in SQL.

Specifying constraints in SQL

CHECK clauses are placed at the end of the CREATE TABLE statement. These are row-based constraints. They are applied to each row individually and whenever a row is inserted or modified.
CHECK (Dept_create_date $<=$ Mgr_start_date);
CREATE ASSERTION can be used for more general constraints.

Basic retrieval queries in SQL

Retrieval makes use of the SELECT statement. The basic structure of the SELECT statement is:

```
SELECT <attribute list>
FROM <table list>
WHERE <condition>;
```

A basic query in SQL can have up to 4 clauses:

```
SELECT <attribute list>
FROM <table list>
[ WHERE <condition> ]
[ ORDER BY <attribute list> ];
```

Basic retrieval queries in SQL

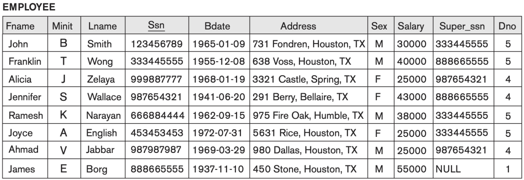
Substring matching is possible, for example:

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE Address LIKE ’%Houston,TX%’;
```

% means 0 or more characters and _ represents a character, for example:

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE Bdate LIKE ’_ _ 7 _ _ _ _ _ _ _’;
```

Basic retrieval queries in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Q0 - Fig 6.3 (a):

```
SELECT Bdate, Address
FROM EMPLOYEE
WHERE Fname = 'John' AND Minit = 'B' AND Lname = 'Smith';
```

Basic retrieval queries in SQL

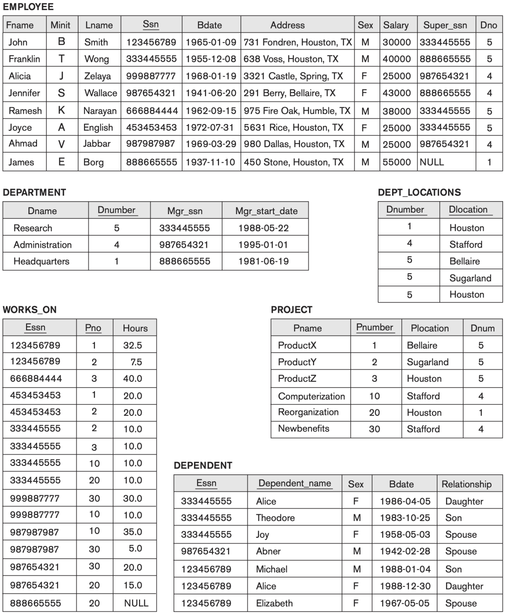
Q1 - Fig 6.3 (b):

```
SELECT Fname, Lname, Address
FROM EMPLOYEE, DEPARTMENT
WHERE Dname = ’Research’ AND Dnumber = Dno;
```

If the attribute representing the department number was specified as Dnumber in all tables, it would be necessary to qualify which Dnumber was being referred to. Q1 would change as follows:
Q1:

```
SELECT Fname, Lname, Address
FROM EMPLOYEE, DEPARTMENT
WHERE Dname = ’Research’ AND
              DEPARTMENT.Dnumber = EMPLOYEE.Dnumber;
```

Figure 5.6
One possible database state for the COMPANY relational database schema.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

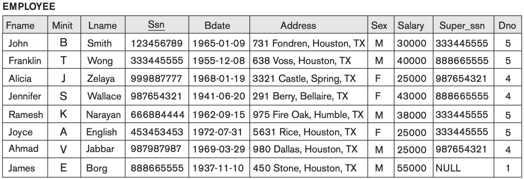
PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Basic retrieval queries in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Q8 - Fig 6.3 (d):

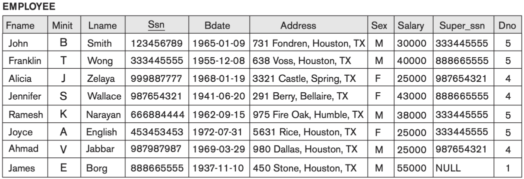
```
SELECT E.Fname, E.Lname, S.Fname, S.Lname
FROM EMPLOYEE AS E, EMPLOYEE AS S
WHERE E.Super_ssn = S.Ssn;
```

Q9 - Fig 6.3 (e):

```
SELECT Ssn
FROM EMPLOYEE;
```

The attribute of the table given in Fig 6.3 (e) is not correct for this query.

Basic retrieval queries in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Q10 - Fig 6.3 (f):

```
SELECT Ssn, Dname
FROM EMPLOYEE, DEPARTMENT;
```

Q1C - Fig 6.3(g)

```
SELECT *
FROM EMPLOYEE
WHERE Dno = 5;
```

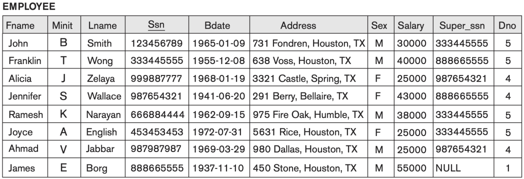
The keyword DISTINCT can be used in the SELECT clause to produce rows with no duplicates. For example:

```
SELECT DISTINCT Salary
FROM EMPLOYEE;
```

INSERT, DELETE and UPDATE in SQL

- INSERT is used to add a Single tuple into a relation
- DELETE removes a tuple(s) from a relation
- UPDATE modifies attribute values in one or more selected tuples

INSERT, DELETE and UPDATE in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

```
INSERT INTO EMPLOYEE
VALUES ( ’Richard’, ’K’, ’Marini’, ’653298653’,
’1962-12-30’, ’98 Oak Forest, Katy, TX’, ’M’,
37000, ’653298653’, 4 );
```

```
INSERT INTO EMPLOYEE (Fname, Lname, Dno, Ssn)
VALUES (’Richard’, ’Marini’, 4, ’653298653’);
```

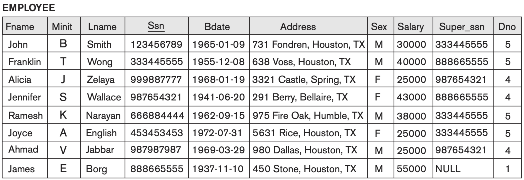
INSERT, DELETE and UPDATE in SQL

Create a table and populate it from other tables:

```
CREATE TABLE WORKS_ON_INFO
( Emp_name VARCHAR(15),
Proj_name VARCHAR(15),
Hours_per_week DECIMAL(3,1) );
```

```
INSERT INTO WORKS_ON_INFO ( Emp_name, Proj_name,
                            Hours_per_week)
SELECT E.Lname, P.Pname, W.Hours
FROM PROJECT P, WORKS_ON W, EMPLOYEE E
WHERE P.Pnumber = W.Pno AND W.Essn = E.Ssn;
```

INSERT, DELETE and UPDATE in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Zero, one or more tuples can be deleted by a single DELETE command

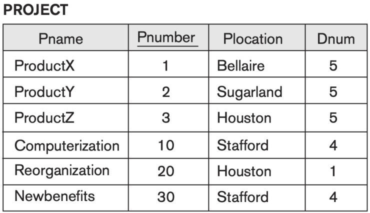
```
DELETE FROM EMPLOYEE
WHERE Lname = ’Brown’;

DELETE FROM EMPLOYEE
WHERE Ssn = ’123456789’;

DELETE FROM EMPLOYEE
WHERE Dno=5;

DELETE FROM EMPLOYEE;;
```

INSERT, DELETE and UPDATE in SQL



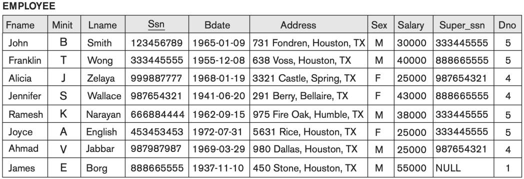
PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

UPDATE uses the SET clause to specify the attributes to be modified

```
UPDATE PROJECT
SET Plocation = 'Bellaire', Dnum = 5
WHERE Pnumber = 10;
```

INSERT, DELETE and UPDATE in SQL



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

UPDATE uses the SET clause to specify the attributes to be modified

```
UPDATE EMPLOYEE
SET Salary = Salary * 1.1
WHERE Dno = 5;
```