SQL

• SQL = “Structured Query Language”
• Standard query language for relational DBMSs
• History:
  Developed at IBM in late 70s
  1\textsuperscript{st} standard: SQL-86
  2\textsuperscript{nd} standard: SQL-92
  3\textsuperscript{rd} standard: SQL-99 or SQL3, well over 1000 pages

  “The nice thing about standards is that you have so many to choose from!”

  -Andrew S. Tannenbaum
SQL

Consists of two parts:

• **Data Definition Language (DDL)**
  Allows the specification of the database schema

• **Data Manipulation Language (DML)**
  Allows the specification of queries & insert/update/delete statements
SQL data definition language
SQL Data Definition Language (DDL)

- Allows the specification of the database schema
  a set of relations with information about each relation

- Schema information:
  - The schema of each relation
  - The domain of values associated with each attribute
  - Integrity constraints

- Other information one can specify:
  - The set of indices to be maintained for each relation
  - Security and authorization information for each relation
  - The physical storage structure of each relation on disk
CREATE TABLE Command

- Used to define a relation
- Syntax:
  ```
  CREATE TABLE relationName
  (attrName1 Domain1,
   ...
   attrNamenn Domainn
  (integrity-constraint1),
  ...,
  (integrity-constraintn))
  ```

- Example:
  ```
  CREATE TABLE branch
  (branch_name char(15) not null,
   branch_city char(30),
   assets integer)
  ```
Domain Types in SQL

- **char(n)**
  Fixed length character string, with user-specified length $n$

- **varchar(n)**
  Variable length character strings, with user-specified maximum length $n$

- **int**
  Integer (a finite subset of integers that is machine-dependent)

- **smallint**
  Small integer (a machine-dependent subset of the integer domain type)
Domain Types in SQL

• **numeric**(p, d)
  Fixed point number, with user-specified precision of p digits, with d digits to the right of decimal point

• **real, double precision**
  Floating point and double-precision floating point numbers, with machine-dependent precision

• **float**
  Floating point number, with user-specified precision of at least n digits

and others…
CREATE TABLE Command

• Can be used to also specify:
  - Primary key attributes (PRIMARY KEY keyword)
  - Secondary keys (UNIQUE keyword)
  - Referential integrity constraints/foreign keys (FOREIGN KEY keyword)

• Example:

  CREATE TABLE DEPT
  (  DNAME      VARCHAR(10) NOT NULL,
      DNUMBER   INTEGER NOT NULL,
      MGRSSN    CHAR(9),
      MGRSTARTDATE CHAR(9),
      PRIMARY KEY (DNUMBER),
      UNIQUE (DNAME),
      FOREIGN KEY (MGRSSN) REFERENCES EMP )

Primary key declaration on an attribute automatically ensures not null in SQL-92 onwards, but it needs to be explicitly stated in SQL-89
DROP TABLE Command

• Used to remove a relation & its *definition*
  The relation can no longer be used in queries, updates, or any other commands since its description no longer exists

• Syntax:
  `DROP TABLE relationName`

• Example:
  `DROP TABLE branch`
ALTER TABLE Command

• Used to add/drop attributes from a relation

• Add attribute syntax:
  ALTER TABLE relationName ADD attribName attribDomain
  All tuples in the relation are assigned null as the default value of the new attribute

• Drop attribute syntax:
  ALTER TABLE relationName DROP attribName
  Dropping of attributes not supported by many DBMSs
ALTER TABLE Command

• Since new attribute will have NULL values right after the ALTER command is executed, the NOT NULL constraint is not allowed for such an attribute.

• Example:

  `ALTER TABLE employee ADD job varchar(12)`

• The database users must still enter a value for the new attribute JOB for each EMPLOYEE tuple. This can be done using the UPDATE command.
Integrity Constraints

• Guard against accidental damage to the database by ensuring that authorized changes to the database do not result in a loss of data consistency.

• Examples:
  - A savings account must have a balance greater than $10,000.00
  - A salary of a bank employee must be at least $6.00 an hour
  - A customer must have a (non-null) phone number
SQL Integrity Constraints

• On single relations:
  - not null
  - primary key
  - unique
  - check(P), where P is a predicate

• On multiple relations:
  - foreign key
NOT NULL Constraint

- Specifies that an attribute does not accept null values
- Can be specified as part of:
  - The definition of an attribute in the CREATE TABLE statement
    e.g. `CREATE TABLE branch
           (branch_name CHAR(15) NOT NULL, ...)
  - The definition of a domain
    (i.e., a “type” that can be used where a type is needed)
    e.g. `CREATE DOMAIN Dollars numeric(12, 2) NOT NULL`
UNIQUE Constraint

• Specifies that a set of attributes form a candidate key
• Syntax:
  \texttt{UNIQUE (AttrName}_1, \ldots, \texttt{AttrName}_n)\texttt{)}

• Candidate keys are permitted to be null (in contrast to primary keys)
CHECK Clause

- Enforce a predicate (condition)
- Syntax:
  CHECK (Predicate)
- Example:
  Ensure that the values of the assets are non-negative

CREATE TABLE branch
  (branch_name char(15),
   branch_city char(30),
   assets integer,
   primary key (branch_name),
   CHECK (assets >= 0) )
CHECK Clause

- Can be also used to constrain domains
  e.g., `CREATE DOMAIN hourly_wage numeric (5,2)`
  `CONSTRAINT value_test CHECK (value >= 4.00)`

- Can be named
  (useful to indicate which constraint an update violated)
  e.g., `CREATE DOMAIN hourly_wage numeric (5,2)`
  `CONSTRAINT value_test CHECK (value >= 4.00)`
Referential Integrity

• Ensures that a value that appears in one relation for a given set of attributes also appears for a set of attributes in another relation.

• Example:
  If “La Jolla” is a branch name appearing in one of the tuples in the account relation, then there exists a tuple in the branch relation for branch “La Jolla”.

Referential Integrity

• In the CREATE TABLE statement we can use:
  - The PRIMARY KEY clause to list primary key (PK) attributes.
  - The UNIQUE KEY clause to list candidate key attributes
  - The FOREIGN KEY clause to list foreign key (FK) attributes and the name of the relation referenced by the FK. By default, a FK references PK attributes of the referenced table.
Referential Integrity Example

```sql
create table customer
    (customer_name  char(20),
     customer_street char(30),
     customer_city   char(30),
     primary key    (customer_name ))

create table branch
    (branch_name    char(15),
     branch_city    char(30),
     assets         numeric(12,2),
     primary key    (branch_name ))
```
Referential Integrity Example

```sql
create table account
    (account_number char(10),
    branch_name    char(15),
    balance        integer,
    primary key (account_number),
    foreign key (branch_name) references branch )

create table depositor
    (customer_name char(20),
    account_number char(10),
    primary key (customer_name, account_number),
    foreign key (account_number ) references account,
    foreign key (customer_name ) references customer )
```