3 SQL语句

- SQL语言有以下几个部分
  - 数据定义语言（DDL，Data-Definition Language）：定义关系模式、删除关系、修改关系模式
  - 数据操纵语言（DML，Data-Manipulation Language）：查询信息，插入、删除、修改元组
  - 完整性
  - 视图定义
  - 事务控制
  - 嵌入式SQL和动态SQL
  - 授权
- create table
  - primary key (A1, ..., An)【主键，唯一的】
  - foreign key (Am, ..., An) references r 【外键，r是另一个表】
- drop table
- alter table
  - alter table r add A
  - alter table r drop A
- 查询语句 select
  - 基本结构 select from where
    - 运算次序
      - From (笛卡儿积) → Where (选择) → Select (投影)
      - From (笛卡儿积) → [Where (选择)] → [Group By (分组)] → [Having (限定分组)] → [Select (投影, 或统计)] → [Order By (结果排序)]
    - select **distinct**(删除重复) / all(不删除重复)
    - ***
      - select * 选择所有属性
      - 关系名.*表示某个关系的所有属性
    - old-name **as** new-name
      - as用在from嵌套子查询建立一个新表之后时，as新表名（属性名1，属性名2…）
      - Where子句中不能使用Select子句中更名后的新属性名
    - select distinct 等价于投影运算
  - where
    - where ... **and ... or ... not**

    - **between ... and ...**
    - **like** '%data%'，反斜杠\可做%的转义字符
    - 如果 where 子句谓词的结果为未知，则将其视为 false
- from 子句：一个关系在From子句中多次出现时，从第二次开始必须重命名
- 集合运算：**自动消除重复项，若要保留后面须跟all**
    - **union**(并)
    - **intersect**(交)
    - **except**(差)
- **null**：is null
    - and : *(true* **and** *unknown)* = *unknown*,
      (false **and** *unknown)* = *false*,
      (unknown **and** *unknown)* = *unknown*
    - or: *(unknown* **or** *true)* = *true*,
      (unknown **or** *false)* = *unknown*
      (unknown **or** *unknown)* = *unknown*
- 聚合函数：**一般用在select后**，对列的多个元组进行操作返回一个值。
    - **avg**
    - **min/max**
    - **sum**
    - **count**（多少个元组）
- 扩展基本结构
    - **group by** 聚集运算：聚合函数之外的select子句中的属性必须出现在group by列表中
    - **having**：只能用在group by之后，对应一组；where 对应一行，在形成组之前
        - Where子句中的条件用于限定元组，施加在单个元组上。不满足条件的元组将不会参与到下一步的分组运算(Group By子句)或投影运算(Select 子句)等
        - **Having**子句中的条件用于限定Group By子句产生的各个分组，施加在整个分组。不满足条件的分组，将不会参与下一步的统计运算(Select子句)
    - **order by** 对结果排序，在Select子句得出结果后，对结果进行排序
        - **desc** 降序
        - **asc** 升序
    - 嵌套子查询
        - 集合比较，用于where
            - **in/not in**:在/不在集合中
            - **some**：F <comp> some r 等价于 t ∈r such that (F <comp> t)
                - Where <comp> can be: ＜， ＞，≤，≥，≠
                - = some 恒等于 in；但是≠some 不恒等于not in

- **all**：对all之后的集合中所有元素都满足；F <comp> all r 等价于t ∈r (F <comp> t)
  - ≠ all 恒等于 not in；但是 = all不恒等于in
- **exists**：如果参数子查询为非空，则exists返回true
- **unique**：测试是否存在重复元组：无重复则返回true

- 用于from的子查询
- **with**：with子句提供了一种定义临时关系的方法，该临时关系的定义仅适用于with子句所在的查询。会放在select之前，相当于定义了新的表，之后被用到

```
with max_budget (value) as
        (select max(budget)
         from department)
select department.name
from department, max_budget
where department.budget = max_budget.value;
```

- 标量子查询：标量子查询是在需要**单个值时**使用的子查询，返回多个值时产生错误
  - 例子：列出所有部门以及每个部门的 讲师人数
    - select dept_name,
      (select count(*)
      from instructor
      where department.dept_name = instructor.dept_name)as
      num_instructors from department;

- 修改数据库
  - 从给定关系中删除元组。
    - 删除整个关系：**delete from** instructor
    - 删除具有特定属性的值的元组：delete from … where
    - 删除满足条件的所有元组: delete from … where…in
  - 在给定关系中插入新元组
    - **insert into** <table> **values('...', '...')**
    - 把某个表里满足条件的元组插入另一个表里：insert into <table> select...from..where...
  - 更新给定关系中某些元组中的值
    - **update** <table> set <属性> = (**case**) 分情况设定
      - 例： update instructor
        set salary = **case** **when** salary <= 100000 **then** salary * 1.05
        else salary * 1.03 **end**

以上内容整理于 幕布文档