[1. IDE 1](#_Toc480495709)

[Navicat的使用 1](#_Toc480495710)

[2. 基本命令 1](#_Toc480495711)

[3. 存储程序 1](#_Toc480495712)

[json写入数据表中 1](#_Toc480495713)

[4. 事务处理 1](#_Toc480495714)

[Key Point 2](#_Toc480495715)

[1. difference between using join and from attached several tables name 2](#_Toc480495716)

# 1. IDE

## Navicat的使用

ctrl+q 打开查询窗口

ctrl+/ 注释

ctrl +shift+/ 解除注释

ctrl+r 运行选中的SQL语句

ctrl+l 删除选中行内容

ctrl+d 复制当前行内容并粘贴到下一行

ctrl+w 关闭当前查询窗口

# 2. 基本命令

$ilike

左连接：

Left JOIN

SELECT \* FROM auth.tokens as at LEFT JOIN auth.accounts as aa on at.id=aa.id

# 存储程序

## json写入数据表中

参考：<https://www.postgresql.org/docs/9.3/static/functions-json.html>

# 4. 事务处理

操作原子性

Window function：

语法：

在现有数据下，执行某个函数，将相应的结果添加显示在新列。如何计算每个部门的平均工资。

Select depname, empno, salary, avg(salary) over (partition by depname) from empsalary

每个window function其后总会跟随over子句，用于决定window function在执行操作时，各行如何被划分成组。

关键字partition by用于分组。但是order by 和 partition by某些情况下可以被忽略。

同一个query中可以执行多个 window function，但是其作用的数据都是同一个virtual table

Window function只存在select和order by从句中。

注意：

Window function相结合的是window frame(分区)，window function的执行作用域就是window frame， 默认是whole rows，即over为空的情况。

# Key Point

## 1. difference between using join and from attached several tables name

---left join：给出所有左边的值和部分右表的值

---right join：给出所有右表的值和部分左表的值

---outer join：

---inner join: 取兩個表的交集

---full join：



Postgresql 🡨🡪 js 🡨🡪 sequelize

对于表属性created\_at、delete\_at、updated\_at字段的添加与否。

当使用sequelize进行model定义时，如sequelize.define() 会提供表结构、表字段说明、以及表额外说明。如freeTableName：true。表示表明与model名一致。