

1. 介绍

云开发提供了一个 JSON 数据库，顾名思义，数据库中的每条记录都是一个 JSON 格式的对象。

一个数据库可以有多个集合（相当于关系型数据中的表），集合可看做一个 JSON 数组，数组中的每个对象就是一条记录，记录的格式是 JSON 对象。

这样的话，数据库的存储也不用考虑了，直接提供了一个类似于 MongoDB一样的数据库，而且免费的存储空间达到了 2G，一般的项目足够使用了。

关系型数据库和 JSON 数据库的概念对应关系如下表：

关系型	文档型
数据库 database	数据库 database
表 table	集合 collection
行 row	记录 record / doc
列 column	字段 field

2. 云数据库操作

注意：操作数据库需要先创建集合名称

The screenshot shows the WeChat Developer Tools (微信开发者工具) interface. At the top, there is a toolbar with various icons for simulation, editing, debugging, and publishing. The 'Cloud Development' icon is highlighted with a red box and has a red arrow pointing down to the 'Cloud Development Console' tab in the bottom navigation bar. The main content area displays the 'Cloud Development Console v1.4.8' interface. On the left, there is a sidebar with tabs for 'Operational Analysis', 'Database', 'Storage', 'Cloud Functions', 'Cloud Hosting', and 'More'. The 'Database' tab is selected. In the center, there is a 'Resource Usage' section with a chart titled 'Resource Usage' showing data from July 3, 2021, to August 3, 2021. The chart includes metrics for Database Capacity (0 MB / 2 GB), Daily Database Read Requests (0 times / 500 times), Storage Capacity (0 MB / 5 GB), Monthly CDN Traffic (0 Byte / 1 GB), and Monthly Cloud Function Resource Usage (0 GBs / 1000 GBs).



效果



1.新建一个db页面

2.初始化数据库

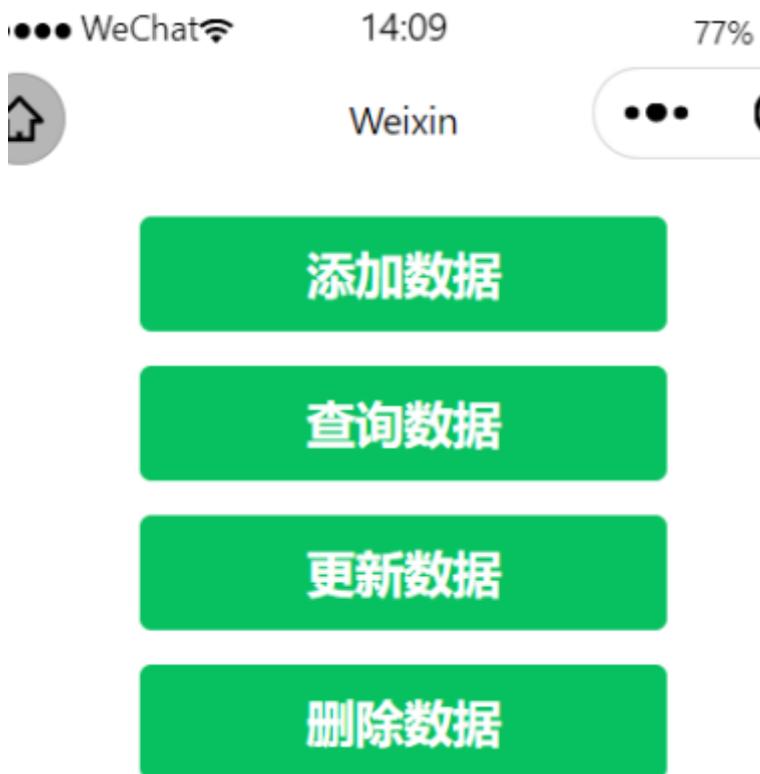
```
wx.cloud.init();
const db = wx.cloud.database();
```

```
// pages/db/db.js
wx.cloud.init()
const db=wx.cloud.database()
const _=db.command;
```

/*
 * 页面的初始数据

3.设置界面

db.wxml



```
<button type="primary" bind:tap="add">添加数据</button>
<button type="primary" bind:tap="get">查询数据</button>
<button type="primary" bind:tap="update">更新数据</button>
<button type="primary" bind:tap="delete">删除数据</button>
```

db.wxss

```
button{
    margin-top: 30rpx;
}
```

4.添加数据

语法: db.collection('集合名称').add("json格式数据")

```
// 添加数据
add:function(){
  db.collection("gpnudb").add({
    data:{
      name:"游龙",
      age:18,
      sex:'男'
```

```

        }
    })
    .then(res=>{
        console.log(res,'添加数据')
    })
    .catch(err=>{
        console.log(err,'添加数据')
    })
},

```

效果

The screenshot shows the WeChat Cloud Development QuickStart interface. On the left, there's a sidebar with a green '添加' (Add) button highlighted with a red box. The main area shows a file tree under '云开发' with 'db' selected, and a code editor with the following JavaScript code:

```

    /**
     * 添加数据
     */
    add:function(){
        db.collection("cloud").add({
            data:{
                name: '游龙',
            }
        })
    }
}

```

The code editor has a yellow warning bar at the bottom: '【sitemap 索引情况提示】根据 sitemap 的规则(0), 当前页面 [pages/db/db] 将被索引'.

On the right, the 'Console' tab of the DevTools shows the result of the insertion:

```

    > {_id: "cc9f30fe5fcefc3000a15bf67bb9150", errMsg: "collection.add:ok"
      _id: "cc9f30fe5fcefc3000a15bf67bb9150"
      > __proto__: Object
    }

```

Below the code editor, the Cloud Database management interface shows a collection named 'cloud' with one document inserted:

字段	值
name	游龙
sex	男
age	18
_id	cc9f30fe5fcefc3000a15bf67bb9150
_openid	os1n64jv_zjB0wbNgP0oX_Df3yms

5. 查找数据

语法: db.collection('集合名称').where("查询条件也使用json格式").get()

```

db.collection('cloud').where({name: 'cloud'}).get().then(res => {
    console.log(res, '数据查询')
}).catch(err => {
    console.log(err, '数据查询')
})

```

[查所有数据](#)

The screenshot shows the WeChat Cloud Development Tools interface. On the left, there's a preview window for a mobile application with two green buttons labeled "添加" (Add) and "查询" (Query). The "查询" button is highlighted with a red border. On the right, the "Resource Manager" sidebar shows a file tree under "miniprogram/pages/db". A code editor window titled "db.js" is open, displaying the following JavaScript code:

```
10
11  },
12 /**
13 * 查询数据库
14 */
15 get:function(){
16     // 查询所有
17     db.collection('cloud').get().then(res => {
18         console.log(res);
19     }).catch(err => {
20         console.log(err)
21     });
22 },
```

效果

The screenshot shows the browser developer tools' "Elements" tab with the "Console" panel open. The output shows the results of a database query, which is an array of two objects. The objects have properties: age, name, sex, _id, and _openid. The first object corresponds to the "游龙" user and the second to the "小龙女" user.

```
▼ {data: Array(2), errMsg: "collection.get:ok"} ⓘ db.js:18
  ▼ data: Array(2)
    ▼ 0:
      age: 18
      name: "游龙"
      sex: "男"
      _id: "cc9f30fe5fcfef6c3000a15bf67bb9150"
      _openid: "os1n64jv_zjBjwbNgP0oX_Df3yms"
      ▶ __proto__: Object
    ▼ 1:
      age: 18
      name: "小龙女"
      sex: "男"
      _id: "846bf2595fcfef77200092cf2f684fe5"
      _openid: "os1n64jv_zjBjwbNgP0oX_Df3yms"
      ▶ __proto__: Object
    length: 2
    nv_length: (...)

  ▶ __proto__: Array(0)
  errMsg: "collection.get:ok"
  ▶ __proto__: Object
```

根据指定条件查询

name 等于小龙女

The screenshot shows the WeChat Mini Program development interface. On the left, the 'Resource Manager' sidebar displays the project structure under 'miniprogram'. A red box highlights the 'db' folder and its 'db.js' file. The main editor window shows the 'db.js' code:

```
10
11 },
12 /**
13 * 查询数据库
14 */
15 get:function(){
16     // 查询 name等于小龙女
17     db.collection('cloud').where({
18         name:'小龙女'
19     }).get().then(res => {
20         console.log(res);
21     }).catch(err => {
22         console.log(err)
23     });
24 }/*
```

效果

The screenshot shows the developer tools debugger. The 'Console' tab is selected. A yellow warning message at the top states: '⚠ [sitemap 索引情况提示] 根据 sitemap 的规则[0], 当前页面 [pages/db/db] 将被索引'. The console output shows the result of the database query:

```
⚠ [sitemap 索引情况提示] 根据 sitemap 的规则[0], 当前页面 [pages/db/db] 将被索引
▼ {data: Array(1), errMsg: "collection.get:ok"} db.js:20
  ▼ data: Array(1)
    ▼ 0:
      age: 18
      name: "小龙女"
      sex: "男"
      _id: "846bf2595fce77200092cf2f684fe5"
      _openid: "os1n64jv_zjBjwbNgP0oX_Df3yms"
      ► __proto__: Object
      length: 1
      nv_length: (...)
```

查询年龄大于18岁的用户

The screenshot shows the WeChat Mini Program development environment. On the left, the project structure is displayed with several files and folders highlighted by red boxes:

- miniprogram**: Contains components, images, pages, and db.
- pages**: Contains addFunction, chooseLib, databaseGuide, db, db.json, db.wxml, db.wxss, and deployFunctions.
- db**: Contains db.js.

The current file is **db.js**, specifically the `get` function. The code is as follows:

```
10
11  },
12 /**
13 * 查询数据库
14 */
15 get:function(){
16     // 查询年龄大于18岁的用户
17     const _ =db.command
18     db.collection('cloud').where({
19         age:_gt(18)
20     }).get().then(res => {
21         console.log(res);
22     }).catch(err => {
23         console.log(err)
24     });
}
```

The execution result is shown below the code editor:

```
▼ {data: Array(1), errMsg: "collection.get:ok"} ⓘ
  ▼ data: Array(1)
    ▼ 0:
      age: 19
      name: "游龙"
      sex: "男"
      _id: "cc9f30fe5fce6c3000a15bf67bb9150"
      _openid: "os1n64jv_zjBjwbNgP0oX_Df3yms"
      ▶ __proto__: Object
      length: 1
      nv_length: (...)

      ▶ __proto__: Array(0)
      errMsg: "collection.get:ok"
      ▶ __proto__: Object
```

查询name等于小龙女与age等于18的用户

The screenshot shows the WeChat Mini Program IDE interface. On the left, the resource manager displays the project structure under 'miniprogram/pages/db'. The current file is 'db.js' at the path 'miniprogram > pages > db > db.js'. The code in 'db.js' is as follows:

```

10
11      },
12  /**
13   * 查询数据库
14   */
15  get:function(){
16      const _ =db.command
17      // 查询name等于小龙女与age等于18的用户
18      db.collection('cloud').where(_.and([
19          {name:'小龙女'}, {age:18}
20      ])).get().then(res => {
21          console.log(res);
22      }).catch(err => {
23          console.log(err)
24      });

```

The screenshot shows the developer tools interface with the results of the database query. A warning message at the top states: '[sitemap 索引情况提示] 根据 sitemap 的规则[0]，当前页面 [pages/db/db] 将被索引'.

The expanded data object is:

```

{
  data: [
    {
      age: 18,
      name: "小龙女",
      sex: "男",
      _id: "846bf2595fce77200092cf2f684fe5",
      _openid: "os1n64jv_zjBjwbNgP0oX_Df3yms",
      __proto__: Object
    }
  ],
  errMsg: "collection.get:ok"
}

```

6. 更新数据

语法:

局部更新

```
db.collection('集合名称').doc(记录id).update(json格式数据)
```

替换更新

```
db.collection('集合名称').doc(记录id).set(json格式数据)
```

```

db.collection('cloud').doc(id).update({
  data: {
    name: 'cloud',
    age:18
  }
}).then(res => {
  console.log(res, '数据更新')
}).catch(err => {
  console.log(err, '数据更新')
})

```

```

12  /**
13  * 更新数据
14  */
15 update:function(){
16   db.collection('cloud').doc('cc9f30fe5fce6c3000a15bf67bb9150').update({
17     data:{
18       name:'龙龙'
19     }
20   }).then(res => {
21     console.log(res)
22   }).catch(err => {
23     console.log(err)
24   });
25 },
26 /**

```

Console output (db.js:21):

```

{stats: {...}, errMsg: "document.update:ok"}
  errMsg: "document.update:ok"
  stats: {updated: 1}
  __proto__: Object

```

_id: cc9f30fe5fce6c3000a15bf67b...

_id: 846bf2595fce77200092cf2f6...

+ 添加字段

```

"_id": "cc9f30fe5fce6c3000a15bf67bb9150"
"_openid": "os1n64jv_zjBJwbNgP0oX_Df3yms"
"age": 19
"name": "龙龙" (highlighted with a red circle)
"sex": "男"

```

7. 删除

语法: db.collection('集合名称').doc(记录d).remove()

```
db.collection('cloud').doc(id).remove().then(res => {
    console.log(res, '数据删除成功')
}).catch(err => {
    console.log(err, '数据删除失败')
})
```

The screenshot shows a code editor interface with the following details:

- Left Sidebar (Resource Manager):** Shows a tree structure of a miniprogram project. The path is: 云开发 > miniprogram > pages > db > db.js. The "db" folder and its sub-file "db.js" are highlighted with red boxes.
- Code Editor Area:** The file content is as follows:

```
8     */
9     data: {
10    },
11   },
12   /**
13   * 删除数据
14   */
15  delete:function(){
16      // 删除_id等于cc9f30fe5fce6c3000a15bf67bb9150
17      db.collection('cloud').doc('cc9f30fe5fce6c3000a15bf67bb9150').remove()
18      .then(res => {
19          console.log(res)
20      }).catch(err => {
21          console.log(err)
22      });
23  },
24  /**
25  * 更新数据
26  */
```

The screenshot shows the Network tab of a browser developer tools window. It displays the following information:

- Request URL:** db.js:19
- Request Method:** GET
- Response Headers:** (None listed)
- Response Body (Raw):**

```
{stats: {}, errMsg: "document.remove:ok"}
```
- Response Body (Decoded):**

```
{stats: {}, errMsg: "document.remove:ok"}
```