# مقدمه

## انواع مغایرت در مرج پایگاه داده ها

1. مغایرت یکتا (uniqueness conflict)

insert/insert conflict

1. مغایرت به روز رسانی (update conflict)

update/update conflict

1. مغایرت حذف (delete conflict)

update/delete conflict

delete/update conflict

delete/delete conflict

| insert/insert | دستی |
| --- | --- |
| update/delete | دستی |
| delete/update | دستی |
| update/update | اتوماتیک |
| delete/delete | اتوماتیک |

## جلوگیری از مغایرت uniqueness

1. تعریف رنج مجزا برای هر نود
2. شروع از یک مقدار خاص برای هر نود
3. تعریف دنباله مشترک
   1. تعریف یک سیستم مرکزی که همه از آن یک عدد منحصر به فرد دریافت کنند.
4. تولید دنباله با روش MMR-ready
   1. باید به هر نود یک id مجزا داده شود

# سناریوهای حل مشکل مغایرت و ایجاد پایگاه داده معتبر

## طراحی یک سیستم مرکزی اتوماتیک که همه گره ها موارد unique را از این سیستم دریافت کنند.

* 1. گره ها باید برای هر درج منتظر سیستم مرکزی باشند.
  2. هنوز هم مشکل درست بودن پایگاه داده ها وجود دارد

## تولید دنباله unique با روش هایی مانند MMR-ready

* 1. باید به هر نود یک id مجزا داده شود برای تولید دنباله
  2. مشکل درست بودن پایگاه داده ها وجود دارد

## طراحی با یک سیستم مرکزی و تعیین وضعیت به صورت دستی توسط کاربر

سناریو به این صورت است که هر گره درخواست(insert, update, delete) خود را برای گره مرکزی ارسال می کند. و گره مرکزی بعد از بررسی درخواست توسط کاربر در صورت تایید‍‍ درخواست را ثبت و برای همه گره ها ارسال می کند. و در صورت عدم تایید جواب را برای گره درخواست کننده ارسال می کند.



فرض کنید سیستم دارای یک جدول به صورت زیر باشد.

| tb1 | | |
| --- | --- | --- |
| id | unique\_num | … |
| 1 | 20123 | … |
| 2 | 30213 | … |

برای این جدول یک جدول به صورت زیر ایجاد می شود.

| tb1\_request | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | tb1\_id | unique\_num | … | from | to | act | status | description |
| 5  7  8 | -  2  2 | 30214  3213  30213 | …… | A  B  D | C  C  C | insert  update  delete | test  pending  accepted  rejected |  |

گره A می خواهد یک سطر اضافه کند. این سطر در یک جدول به صورت زیر اضافه می شود.

| tb1\_request | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | tb1\_id | unique\_num | … | from | to | act | status | description |
| 1 |  | 30214 | … | A | C | insert | pending |  |

گره C یک درخواست به صورت زیر از گره A دریافت می کند.

| tb1\_request | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | tb1\_id | unique\_num | … | from | to | act | status | description |
| 10 |  | 30214 | … | A | C | insert | pending |  |

گره C بعد از تایید tb1 , tb1\_request را به صورت زیر تغییر می دهد.

| tb1 | | |
| --- | --- | --- |
| id | unique\_num | … |
| 1 | 20123 | … |
| 2 | 30213 | … |
| 3 | 30214 | … |

| tb1\_request | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | tb1\_id | unique\_num | … | from | to | act | status | description |
| 10 | 3 | 30214 | … | A | C | insert | accepted |  |

و سپس گره C این درخواست را به گره های A , B ,D می فرستد. و در tb1 همه گره ها درج می شود و tb1\_request گره A به صورت زیر به روز رسانی می شود.

| tb1\_request | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | tb1\_id | unique\_num | … | from | to | act | status | description |
| 1 | 3 | 30214 | … | A | C | insert | accepted |  |

### مشکل غیر همزمان بودن ارسال بین گره ها:

* مطمئن شدن از اینکه ارسال به درستی انجام شده است
* مطمئن شدن از اینکه ارسال برای همه گره ها انجام شده است

جدول ارسال درخواست:

| send\_requence | | |  |
| --- | --- | --- | --- |
| id | req\_id | node | status |
| 1 | 1 | A | sending  undone  done |

* در زمان ارسال یک درخواست باید در جدول send\_requence ثبت شود و تا زمان مطمئن شدن از رسیدن به مقصد دوباره ارسال شود.
* در هر گره باید امکان تشخیص تکراری بودن یک درخواست وجود داشته باشد.