Backend Study Guide — Laravel, MySQL, Redis, SQS (Detailed)

Mục tiêu: Nắm cơ chế, biết khi nào dùng, hiểu ưu/nhược điểm và lưu ý vận hành.

Eloquent nâng cao: eager loading (with, loadMissing)

Giảm N+1 queries, tải quan hệ cần thiết. `with` eager trước khi query; `loadMissing` bổ sung sau khi đã có model.

Code Snippet

```
// Eager trước khi query
$orders = Order::with(['customer', 'items.product'])->whereBetween('created_at', [$from,$to])

// B sung quan h thi u
$orders->loadMissing('payments');

// Conditional eager
Order::when($withItems, fn($q)=>$q->with('items'))->get();
```

Ưu điểm

Giảm round-trip DB, cải thiện p95 latency trên trang nhiều quan hệ.

Nhược điểm

Dễ over-fetch nếu eager quá nhiều quan hệ không cần → tăng RAM/serialize time.

Notes / Lưu ý

Ưu tiên chọn lọc fields với `select()` và giới hạn depth. Dùng `::withCount()` nếu chỉ cần đếm.

Eloquent: subquery select & scopes & when()

Subquery để lấy computed column; scopes để tái sử dụng filter; `when()` để điều kiện hoá.

Code Snippet

```
// Subquery select
$lastPayment = DB::table('payments')->selectRaw('MAX(paid_at)')->whereColumn('payments.order_
$orders = Order::query()->select(['id','total','created_at'])
  ->addSelect(['last_payment_at' => $lastPayment])
  ->when($status, fn($q)=>$q->where('status',$status))
  ->latest('id')->paginate(50);

// Scope
class Order extends Model { public function scopeActive($q){ return $q->where('status','active
```

Ưu điểm

Ít join nặng; tái sử dụng filter sạch; `when()` gọn.

Nhược điểm

Subquery có thể không dùng index; cần EXPLAIN.

Notes / Lưu ý

Đảm bảo index (order_id, paid_at).

MySQL 8: Index, EXPLAIN, Keyset Pagination

Tối ưu truy vấn lớn; tránh OFFSET; dùng tuple comparison.

Code Snippet

```
CREATE INDEX idx_tx_created_id ON transactions (created_at, id);
SELECT id, amount, created_at FROM transactions
WHERE (created_at, id) > (?, ?) ORDER BY created_at, id LIMIT 101;
```

Ưu điểm

Latency ổn định; ít CPU hơn OFFSET.

Nhược điểm

Không nhảy trang tự do; cần quản lý cursor.

Notes / Lưu ý

Trả cursor bản ghi cuối; tránh function trên cột index.

Redis: Cache-aside, Lock, Token Bucket

Chiến lược cache & concurrency control.

Code Snippet

```
// Cache-aside
$user = Cache::remember("user:$id", 600, fn()=>User::findOrFail($id));
// Lock
Cache::lock('refresh_token_api_'.$companyId, 60)->block(10, function () {/* critical */});
// Rate limit
RateLimiter::for('partner.push', fn()=> Limit::perMinute(120)->by('partner:'.$partnerId));
```

Ưu điểm

Giảm tải DB; tránh race; kiểm soát lưu lượng.

Nhược điểm

Invalidation khó; TTL/lock sai gây lỗi.

Notes / Lưu ý

Theo dõi cache hit ratio; có circuit breaker downstream.

SQS: Visibility Timeout, Idempotency, DLQ

At-least-once delivery cần idempotency.

Code Snippet

```
public function handle(){
  $key = "job:push:{$this->naturalId}";
  $lock = Cache::lock($key, 120);
  if(!$lock->get()) return;
  try {
    DB::table('partner_payloads')->upsert(
      ['natural_id'=>$this->naturalId, 'payload'=>json_encode($this->data), 'pushed_at'=>now()
      ['natural_id'], ['payload','pushed_at']
    );
  } finally { $lock->release(); }
```

```
}
```

Ưu điểm

Giảm duplicate; có DLQ.

Nhược điểm

Cần theo dõi visibility & DLQ.

Notes / Lưu ý

Visibility > 2x p95; cảnh báo DLQ.

Export/Import lớn: Streaming & Upsert

Xuất CSV streaming; nhập batch + upsert + retry deadlock.

Code Snippet

```
// Streaming CSV
Route::get('/export/csv', function(){ /* ... see previous snippet ... */ });
// Upsert batch
DB::table('tankmonitors')->upsert($batch, ['type','monitor_id'], ['name','product','location',
```

Ưu điểm

TTFB nhanh; nhập an toàn idempotent.

Nhược điểm

Streaming phụ thuộc proxy; upsert cần index đúng.

Notes / Lưu ý

Checkpoint Redis; error logs riêng.