

โอเค เข้าใจเลย — ฉันจะจัด สรุปละเอียด + ตัวอย่างข้อสอบเยอะๆ ให้ตามนี้:
(ถ้าอยากได้แบบ Mindmap หรือ Quiz ฝึกมือพิเศษบอกได้เลยนะ)

สรุปละเอียดก่อนสอบ

1. SQL และ NoSQL (PostgreSQL, MongoDB)

PostgreSQL (SQL)

- **DDL (Data Definition Language)**

ใช้จัดการโครงสร้าง Database เช่น

```
CREATE TABLE students (  
  id SERIAL PRIMARY KEY,  
  name VARCHAR(50)  
);  
DROP TABLE students;
```

- **DML (Data Manipulation Language)**

ใช้จัดการข้อมูล เช่น

```
INSERT INTO students (name) VALUES ('John');  
SELECT * FROM students;  
UPDATE students SET name = 'Mike' WHERE id = 1;  
DELETE FROM students WHERE id = 1;
```

- **Foreign Key และ Constraint**

- ใช้เชื่อมตาราง:

```
CREATE TABLE orders (  
  order_id SERIAL PRIMARY KEY,  
  student_id INT,  
  CONSTRAINT fk_student FOREIGN KEY(student_id) REFERENCES  
  students(id)  
);
```

- **CASCADE DELETE** ลบข้อมูลใน child table อัตโนมัติเมื่อลบ parent:

```
ON DELETE CASCADE
```

- **CHAR vs VARCHAR**

- **CHAR(n)**: กินเนื้อที่ fix ความยาว n ตัวอักษร (เหลือเต็ม space)
- **VARCHAR(n)**: เก็บแค่เท่าที่ใช้จริง สูงสุด n ตัวอักษร

- **CHECK Constraint**

- ใช้บังคับเงื่อนไขข้อมูล:

```
CREATE TABLE employees (  
  id SERIAL,  
  age INT CHECK (age >= 18)  
);
```

- **Default Port ของ PostgreSQL: 5432**

- **แสดงตารางทั้งหมด**

- `\d` หรือ `\dt` (ใน psql)

- **วัดประสิทธิภาพ Query**

- ใช้ `EXPLAIN` หรือ `EXPLAIN ANALYZE`

MongoDB (NoSQL)

- **Schema-less**: ข้อมูลไม่จำเป็นต้องมีโครงสร้างเดียวกันทุก document

- **เก็บข้อมูลแบบ JSON/BSON**

- **Sorting**

- `.find().sort({ field: 1 })` (1 = ASC, -1 = DESC)

- **Embedding vs Referencing**

- **Embedding**: ยัดข้อมูลย่อยลงไปเลย (เหมาะกับ one-to-few)
- **Referencing**: เก็บแยกแล้วลิงก์ด้วย id (เหมาะกับ one-to-many ขึ้นไป)

- **Query Comparison**

- `gte` = greater than or equal

```
db.students.find({ age: { $gte: 18 } })
```

- **\$lookup**: ทำ JOIN

```
db.orders.aggregate([
  { $lookup: {
    from: "products",
    localField: "product_id",
    foreignField: "_id",
    as: "product_info"
  }}
])
```

- **\$project**: คล้าย SELECT
- **\$unwind**: แยก Array
- **Default Port MongoDB: 27017**

2. Indexing (12 คะแนน)

B+ Tree Index

- ใช้บ่อยใน Database จริง
- เหมาะกับ **query range** เช่น age > 18
- สนับสนุน Equality + Range Query

Clustered vs Unclustered

- **Clustered**: ข้อมูลจริงเรียงตาม index (primary key)
- **Unclustered**: Index แยกจากข้อมูลจริง

ตัวอย่าง index ที่ควรใช้

Query	Index ที่ควรใช้
หานักศึกษาที่จองห้องเดียว	unclustered B+ tree บน (idห้อง, idนศ)
หาห้องที่นักศึกษาหมายเลข 4 จอง	unclustered B+ tree บน (idนศ, idห้อง)

อื่นๆ

- **Secondary Index** ต้องเป็น **dense**
- **Root node** ของ **B+ Tree** ต้อง **unique**
- **Hash index** search เวลาเฉลี่ยเท่าเดิมทุกกรณี
- อย่าสร้าง **index** เยอะถ้า **database** มีการเปลี่ยนแปลงบ่อย

3. Stored Routine และ Window Function

Stored Routine

- **Stored Procedure**: ชุด SQL ทำหลายขั้นตอน

- **Function:** รับ param และ return ค่าเดียว
- **Trigger:** ทำงานอัตโนมัติ เมื่อมี event เช่น INSERT, UPDATE

ตัวอย่าง

```
CREATE FUNCTION get_total_order(student_id INT)
RETURNS INT AS $$
BEGIN
    RETURN (SELECT COUNT(*) FROM orders WHERE student_id = student_id);
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER after_order_insert
AFTER INSERT ON orders
FOR EACH ROW
EXECUTE FUNCTION update_order_count();
```

Window Function

- ใช้ **OVER()** สำหรับ "แบ่งกลุ่ม" ข้อมูล
- ตัวอย่าง:

```
SELECT name, salary,
       RANK() OVER (ORDER BY salary DESC) AS salary_rank
FROM employees;
```

- Aggregate ปกติสรุปหลายแถว, Window function ไม่สรุป แต่คำนวณต่อแถว

ตัวอย่างข้อสอบ (เยอะๆ)

Part 1: SQL/PostgreSQL

1. เขียน **SQL** สร้างตาราง **student** มี **id**, **name** และ **age** ที่ต้อง ≥ 18
2. ลบตารางชื่อ **student**
3. **Query** หานักศึกษาที่อายุมากกว่า 20 ปี เรียงตามชื่อ
4. สร้าง **Foreign Key** เชื่อมระหว่างตาราง **orders** กับ **students**
5. ลบข้อมูลใน **orders** ทั้งหมดถ้า **student** ถูกลบ

Part 2: MongoDB

1. **Query** หานักเรียนที่มีอายุมากกว่าหรือเท่ากับ 18 ปี
2. เขียน **Aggregation** ใช้ **\$lookup** เพื่อ **JOIN** **students** กับ **orders**

3. ใช้ \$project เลือกเฉพาะชื่อกับอายุ
4. เรียงข้อมูลจากคะแนนมากไปน้อย
5. แยก array ของ hobbies ออกมาเป็น document เดี่ยวๆ

Part 3: Index

1. เลือก Index ที่เหมาะสมสำหรับการ query ข้อมูลช่วงช่วงอายุ 18-25 ปี
2. อธิบายความแตกต่าง Clustered กับ Unclustered index
3. ทำไมไม่ควรสร้าง index เยอะในระบบที่ update บ่อย
4. ให้ข้อมูล 5*5, คำนวนการ search/insert/delete B+ tree เทียบกับ Hash index




Part 4: Stored Routine และ Window Function

1. อธิบาย Stored Procedure, Function, Trigger ต่างกันยังไง
2. สร้าง Function คำนวณจำนวน order ของ student
3. ใช้ Window function หา salary_rank ของพนักงาน

เทคนิคเสริมจำง่ายก่อนสอบ

- "CASCADE" = ลบตามสาย
- "EXPLAIN" = วิเคราะห์ query
- MongoDB = NoSQL, JSON
- PostgreSQL = SQL, DDL, DML
- Index = B+ Tree ดีสำหรับช่วงข้อมูล
- Secondary index = dense
- Stored Procedure = ชุด SQL หลายคำสั่ง
- Window function = มี OVER()

สรุปไฟล์ที่ได้ตอนนี้

-  ข้อมูลสรุปละเอียด
-  ตัวอย่างข้อสอบเยอะ
-  Key Point สำคัญๆ

อยากให้ทำ mini-sheet แบบ 1 หน้ากระดาษ หรือสรุปแบบ Flashcard/Quiz ฝึกมือก่อนไปสอบไหม? 🚀
(ตอบเลย เดี่ยวทำให้อีกชุดฟรี)