Week 0 Querying

- scale : data amount
- frequency : modify
- speed : lookup

Database : A collection of data organized for CRUD

DB Management System : software, interact w/ db.

SQL (Structured Query Language)

- sqlite3 file.db
- .header on
- .mode table / column / list / json ...

※ ctrl - L ⇒ clear

```
SELECT * FROM "table";
```

"column", "col2" ...

- "..." : SQL identifiers for Column name
- '...' : String

※ Column name mispelled ⇒ create data of all rows

odd results

SELECT
LIMIT
WHERE

case sensitive ↑

same

= , != , <> ↔

NOT eg. WHERE NOT sth. = cond.

AND. OR. ( ). >. <. >=. <=

BETWEEN... AND... (Inclusive)

NULL ⇒ IS NULL / IS NOT NULL

LIKE { % any char. eg. '% love %'
_ any Single char. (underscore)
Case insensitive

ORDER BY ... ASC / DESC (Int / Alpha)

COUNT. AVG. MIN. MAX. SUM aggregate func

eg. AVG("rating"), COUNT(*) # rows

ROUND. eg. ROUND(AVG("rating"), 2)

Rename Column col_name AS "name"

DISTINCT eg. SELECT DISTINCT "col-name"

※ { . help
. quit

Week 1 Relating Multiple table.

Relational Database

{ 1 to 1
1 to many
many to many

{ —○— zero
—|— one
—<— many

ER (Entity Relationship) Diagram

eg. 

Author write one book.

Keys { Primary Key. id
Foreign Key

subqueries
nested queries

```
SELECT "title" FROM "books"
WHERE "publisher_id" = (
    SELECT "id" FROM "publishers"
    WHERE "publisher" = 'Fitzcarraldo Editions'
);
```

IN eg. WHERE "id" IN (...)
[ids].

JOIN... ON
eg. "table 1" JOIN "table 2" ON
"table 1"."id" = "table 2"."id"

INNER JOIN ∩ intersect

LEFT OUTER JOIN
RIGHT OUTER JOIN
FULL OUTER JOIN
} Have priorities.
Can have null

NATURAL JOIN 不需要 ON ... id = ... id
直接 assume

Sets { INTERSECT ∩ same # column.

UNION.
A EXCEPT B (A - B)

name & type

Groups
- GROUP BY column_name
- HAVING cond — filter

Week 2 Designing

.schema — show the CREATE stmt matching pattern (that opens the table)

or .schema table_name

※ REAL ⇒ floating #, decimals.

.read sth.sql — # execute queries.

Normalize — Remove redundancy.

CREATE TABLE "name" (attr ...)

- Data Types. ⇒
  - NULL
  - INTEGER
  - REAL
  - TEXT
  - BLOB — Binary Large OBj. (圖像 ...)
- Storage Classes

Type Affinities — Convert into this type.

- TEXT
- NUMERIC — 高精度 eg. 1.00 eg.
- INTEGER

CRUD 常见转换

REAL
BLOB

NUMERIC(10,2)
十位数，二位小数

DROP TABLE

Table Constraints

- PRIMARY KEY ("id")
- FOREIGN KEY ("table_id") REFERENCES "table" ("id")

Column Constraints

- CHECK eg. ("col" > 10) cond
- DEFAULT default value
- NOT NULL
- UNIQUE

Altering Table

eg. Drop table "table"; ← 必加

ALTER TABLE "table" +

⇒
```
RENAME TO "new_name";
ADD COLUMN "name" constraint;
RENAME COLUMN "a" TO "b";
DROP COLUMN "name";
```

eg. CHECK ("col" > 10)

("col" IN (1, 2, 3))

Week 3 Writing

PK : No need for "id"

CREATE ⇒ INSERT INTO table (column ...)

VALUES

(value 0, ...),

(value 1, ...);

Insert more data

顺序、数量相同.

※ constraint (Not Null, Unique) fail ⇒ Runtime Error.

.import --csv --skip 1 file.csv table

skip the 1st header row

.import --csv file.csv table


```
INSERT INTO table0 (column0, ...)
SELECT column0, ... FROM table1;
```

两个 table Column 数量对应.

DELETE FROM table WHERE condition;

Foreign Key Constraints

FOREIGN KEY ("key") REFERENCES "table"("id")

ON DELETE + RESTRICT (Not allowed to del)

NO ACTION

SET NULL
SET DEFAULT
CASCADE ✡ - 起删.

UPDATE table SET column0 = value0, ...
WHERE condition;

也可以用 subquery.

✡ SQLite Scalar function

trim(col) # trim white space.

upper(col)
lower(col)

Triggers show up in transaction. keep logs

any table

CREATE TRIGGER name
BEFORE / AFTER
INSERT ON table
UPDATE OF column ON table
DELETE ON table
FOR EACH ROW
BEGIN
... ; ← stmts.
END;

✡ before/after
⇒ OLD.column
NEW.column

备份的 Insert / update 都可. keep log.

Week 4 Viewing.

View: A virtual table defined by a query.

- Simplifying
- Aggregating
- Partitioning
- Securing

不存在 ⇒ 自动补齐.

eg. 'Anonymous' as 'col'.

```
CREATE VIEW name AS [SELECT ... ;]
```

(TEMPORARY)
↳ .quit 以后就删了.

Query

CTE (Common Table Expression)

```
WITH name AS (
        SELECT ...
    ), ...
SELECT ... FROM name;
```

可子加.

※ CAN'T MODIFY VIEW.

不能 update / alter view.

Soft Deletion: 加 column = "deleted"
记录是否 deleted.
set default = 0.

```
CREATE TRIGGER name
INSTEAD OF INSERT ON view
FOR EACH ROW WHEN condition
BEGIN
    ...;
END;
```

update table

- TRIGGER & VIEW
  - ⇒ { OLD.col
        NEW.col.

Week 5 Optimizing.

IMDB : internet movie db.

※ , timer on # time future query

Index : A structure used to speed up the retrieval of rows from a table

```
CREATE INDEX name ON table (column...);
EXPLAIN QUERY PLAN SELECT ...;
```

↘ 解释 sqlite 打算 怎么 execute 这个 query. ↘

※ DROP INDEX name;

Covering Index: An index in which queried data can be retrieved from the index itself.

B-Tree: A balanced tree structure commonly used to create an index.

Partial Index: An index that includes only a subset of rows from a table.

CREATE INDEX name ON table (column…)
WHERE condition;

※ du -b file # the usage of file in bytes.

★ VACUUM; # DROP INDEX 以后 VACUUM; bytes 会 返还 给 OS.

Concurrency queries at the same time.

⇓

Transaction: A unit of work in a database

- **A**tomicity can't break down
- **C**onsistency don't vidate constraint
- **I**solation queries won't interfere each other
- **D**urability data remains

```
BEGIN TRANSACTION;
...
```

COMMIT; / ROLLBACK;
(重回上个状态)

Race Conditions : ⇒ Inconsist state

Multiple processors access the same value.

Transfer $30 to Alice	Transfer $30 to Alice	Withdraw $60
Charlie's balance: $30		
	Charlie's balance: $30	
+$30 to Alice		
	+$30 to Alice	
		Alice's balance: $60
		Withdraw $60
-$30 from Charlie		
	-$30 from Charlie	
	CHECK constraint failed	

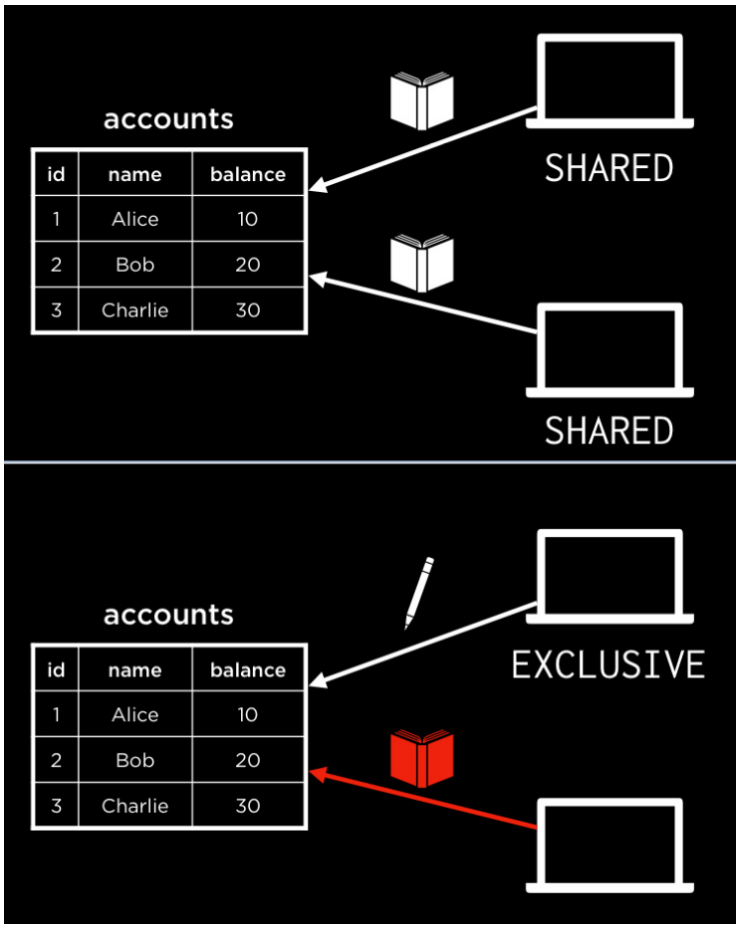
Transfer $30 to Alice	Transfer $30 to Alice	Withdraw $60
Begin transaction		
Charlie's balance: $30		
+$30 to Alice		
-$30 from Charlie		
Commit		
	Begin transaction	
	Charlie's balance: $0	
	+$30 to Alice	
	-$30 from Charlie	
	Rollback	

Locks : ensure transactions are sequential
don't interfere w/ each other

- UNLOCKED
- SHARED
- EXCLUSIVE
- ...

Block transaction.

eg. BEGIN EXCLUSIVE TRANSACTION;



Week 6 Scaling

Scalability : Ability to increase or decrease capacity to meet demand.

MySQL Login: MYSQL

```
mysql -u root -h 127.0.0.1 -P 3306 -p
```

as root (admin) user · host · localhost · port · root pwd

```
SHOW DATABASES;
CREATE DATABASE 'mbta';
USE 'mbta';
```

name

All single quotes

```
DESCRIBE 'mbta';
```

. Schema in table format

MySQL Data Type:

INTEGER

⇓

INT

Data Type	Size (in Bytes)	Minimum Value (Signed)	Maximum Value (Signed)
TINYINT	1	-128	127
SMALLINT	2	-32,768	32,767
MEDIUMINT	3	-8,388,608	8,388,607
INT	4	-2,147,483,648	2,147,483,647
BIGINT	8	-2^{63}	$2^{63} - 1$

Data Type	Size (in Bytes)	Minimum Value (Unsigned)	Maximum Value (Unsigned)
TINYINT	1	0	255
SMALLINT	2	0	65,535
MEDIUMINT	3	0	16,777,215
INT	4	0	4,294,967,295
BIGINT	8	0	$2^{64} - 1$

AUTO_INCREMENT

```
CREATE TABLE `cards` (
    `id` INT AUTO_INCREMENT,
    PRIMARY KEY(`id`)
);
```

Strings ⇒
- CHAR(M) # ONLY 2 char.
- VARCHAR(M) multiple char.

TEXT ⇒
- TINYTEXT
- TEXT
- MEDIUMTEXT
- LONGTEXT

- BLOB
- ENUM
- SET

choose multiple option from the list.

Dates. ⇒
Times
- DATE
- TIME (fsp)
- DATETIME (fsp)
- TIMESTAMP (fsp)
- YEAR

specify decimal digits.

CURRENT_TIMESTAMP

Real Numbers
Floating-Point
Imprecision
Fixed Precision

⇒ DECIMAL(M, D)

M ↓ Full Length
D ↓ Decimal

eg. (5,2) ⇒ (-999.99, 999.99)

※ MUL: multiple value ⇒ Foreign Key
MUST have Proper Types

MySQL Alter Table

ALTER TABLE ...

MODIFY ... ; ⇒ column change

Stored Procedures

similar to TRIGGER

```
CREATE PROCEDURE name(parameters)
BEGIN
...
END;
```

※ IN col type (input)

```
IF, ELSEIF, ELSE
LOOP
REPEAT
WHILE
...
```

USE DB_name; # subsequent query run on this database

SHOW TABLES;

ALTER TABLE ...

ADD COLUMN 'name' TINYINT;

※ Delimiter Change: delimiter // ; 容易混

BEGIN ... END // 中间还用 ;

CALL 'procedure'(); save query + use it

params

PostgresSQL

Double quotes.

INT (Whole number)	SMALLINT	2 bytes	
	INT	4	
	BIGINT	8	
	SMALLSERIAL		AUTO_INCREMENT good for PK.
	SERIAL		
	BIGSERIAL		

```
psql postgresql://postgres@127.0.0.1:5432/postgres

\l          # list databases.
CREATE DATABASE 'mbta';
\c 'mbta'   #  USE database.
\dt         #  show tables.
\d table    #  show particular schema.
\q          #  .quit
```

DateTime Real Number

```
TIMESTAMP(p)        MONEY
DATE                NUMERIC(precision,scale)
TIME(p)
INTERVAL(p)
```

eg. DEFAULT now()

CREATE TYPE name AS ...

```
CREATE TYPE "swipe_type" AS ENUM('enter, …);

CREATE TABLE "swipes" (
    …,
    "type" "swipe_type" NOT NULL,
    "datetime" NUMERIC NOT NULL DEFAULT CURRENT_TIMESTAMP,
    "amount" NUMERIC NOT NULL CHECK("amount" != 0),
    …
);
```

Vertical Scaling: Increasing capacity by increasing a server's computing power. 1 server ↑

Horizontal Scaling: By distributing load across multiple servers. more server

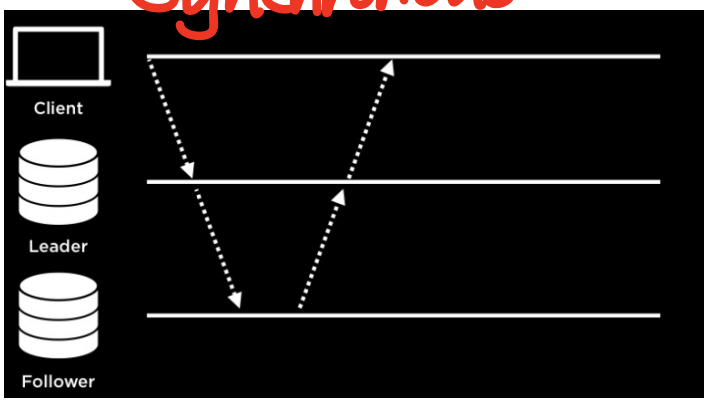
Replication: Keeping copies of a database on multiple servers.

- Single-Leader
- Multi-Leader
- Leaderless

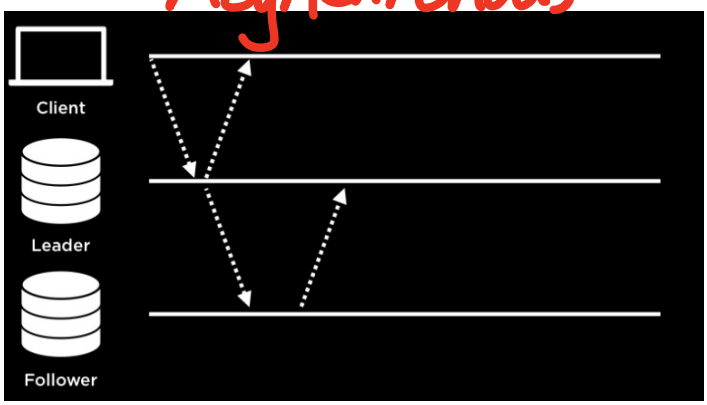
Read Replica: A copy of a db from which data may only be read

- Synchronous — leader wait for follower process
- Asynchronous — NOT Wait

Synchronous



Asynchronous



Sharding 分片. 分区.

Access Control

```
CREATE USER name IDENTIFIED BY passwd;
GRANT privilege, ... TO user;
REVOKE privilege, ... FROM user;
```

> root 给 user

eg. `GRANT SELECT ON table TO user;`

ALL, CREATE, INSERT, SELECT, UPDATE ...

SQL Injection Attacks

eg. WHERE ...

OR '1' = '1'; # Always True.

UNION SELECT * FROM table # 并集

Prepared Statements Clean input

PREPARE name FROM statement;

EXECUTE name USING value 0, ...;

eg. PREPARE 'check'

FROM 'SELECT * FROM table'

WHERE 'id' = ?';

wait for input.

⇒ SET @id = 1;

EXECUTE 'check' USING @id;

Clean the Stmt.

这里 SQL Injection Attack 不起作用：因为 escape SQL query within the variable.