

SQL

Structured Query Language

- Originally developed in the System-R project of IBM (1974)
- Industry standard for relational databases (SQL92 is an ANSI/ISO standard)

Structured Query Language

- Data Definition Language for defining relations, views, integrity constraints, triggers
- Data Manipulation Language for updating, and querying
- Database Control Language for defining access rights, concurrency control, etc....

SQL - Data Definition Language

- Create table
 - Integrity Constraints (next lecture)
- Delete Table
- Alter Table

SQL DDL, creation (simple)

```
CREATE TABLE relation-name  
(attribute-name domain  
[, attribute-name domain ])
```

Example: CREATE TABLE branch
(name varchar(10),
city varchar(20),
director varchar(20),
assets number)

branch	name	city	director	assets

Integrity Constraints – Primary Key

- **Primary Key** is a **set** of attributes which identifies **uniquely** a tuple (i.e., a row in a table)
 - E.g., your **NRIC** or your **email address**
 - The **combination** (name, city) in the branch table
- You cannot have two tuples with the same Primary Key in a table
 - E.g., there cannot be two persons with the **same NRIC**. There cannot be two branches with the **same name in the same city**

SQL DDL, creation example

```
CREATE TABLE branch
( name          varchar(10),
  city          varchar(20),
  director      varchar(20),
  assets        number,
  PRIMARY KEY (name, city) )
```

SQL DDL, Reference to other table

```
CREATE TABLE workfor
( branch_name  varchar(10) ,
  city         varchar(20),
  employee     varchar(20) REFERENCES staff(name) )
```

workfor

branch_name	city	employee
Buona Vista	Singapore	John
Buona Vista	Singapore	Tom
Clementi	Singapore	Tom

staff

name	Address	Tel
Tom	Addr1	21223343
John	Addr2	61223367
Helen	Addr3	97229343

name must be the primary key of *staff*

SQL DDL, deletion

```
DROP TABLE relation_name
```

Example: DROP TABLE branch

SQL DDL, alteration

```
ALTER TABLE relation_name ADD Att Domain
```

Example: ALTER TABLE branch ADD zip INTEGER

branch	name	city	director	assets	zip

```
ALTER TABLE relation_name DROP Att
```

Example: ALTER TABLE branch DROP zip

branch	name	city	director	assets

SQL – Data Manipulation Language

- Insert
- Delete
- Update
- Queries
 - Simple Selections
 - Advanced queries (Aggregations, etc.)
 - Nested Queries
 - Views

SQL DML, insertion (values)

```
INSERT INTO relation_name [(Att [,Att]*)] VALUES (value [,value]*)
```

Example:

```
INSERT INTO branch (name, director, city, assets)
VALUES ('Clementi', 'Ng Wee Hiong',
       'Singapore', 3000000)
```

branch

name	city	director	assets
Clementi	Singapore	Ng Wee Hiong	3000000

SQL DML, insertion (query)

INSERT INTO *relation_name* [(Att [,Att]*)] *query*

Example: INSERT INTO *johor_director* (we assume such a table was created)
 SELECT *director*
 FROM *branch* WHERE *city* = 'Johor Barhu'

branch				johor_director	
name	city	director	assets	director	
Clementi	Singapore	Ng Wee Hiong	3000000	John	
F_branch	Johor Barhu	John	1500000	George	
S_branch	Johor Barhu	George	1200000		

SQL DML, deletion

DELETE FROM *relation_name* [WHERE *qualification*]

Example: DELETE FROM *branch*
 WHERE *city* = 'Jakarta' and *assets* < 1000000

branch			
name	city	director	assets
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Barhu	John	1500000
S_branch	Johor Barhu	George	1200000
Branch_one	Jakarta	Bo Lee	80000
Monas	Jakarta	Agus Arianto	4000000

SQL DML, update

UPDATE *relation_name*
 SET *att* = *expr*
 [WHERE *qualification*]

SQL DML, update example

branch			
name	city	director	assets
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Barhu	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1000000

UPDATE *branch*
 SET *assets* = *assets* * 1.5
 WHERE *city* = 'Kuala Lumpur'

branch			
name	city	director	assets
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Barhu	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1500000

SQL DML, simple query

SELECT [DISTINCT] *target-list*
 FROM *relation-list*
 [WHERE *qualification*]

SQL DML, simple query example

SELECT *
 FROM *work_for*

work_for		
branch_name	city	employee
Clementi	Kuala Lumpur	Yu Fei
Clementi	Singapore	Ng Wee Hiong
Clementi	Singapore	Peter Ho
Clementi	Singapore	Jean Do
Monas	Jakarta	Agus Arianto
Monas	Jakarta	Reza Santi

SQL DML, simple query example

branch

name	city	director	assets
Branch_one	Jakarta	Bo Lee	80000
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Barhu	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1000000
Monas	Jakarta	Agus Arianto	4000000
S_branch	Johor Barhu	George	1200000

```
SELECT name, city
FROM branch
```

branch	name	city
	Branch_one	Jakarta
	Clementi	Singapore
	F_branch	Johor Barhu
	KL_branch	Kuala Lumpur
	Monas	Jakarta
	S_branch	Johor Barhu

SQL DML, simple query example

name	city	director	assets
Branch_one	Jakarta	Bo Lee	80000
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Barhu	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1000000
Monas	Jakarta	Agus Arianto	4000000
S_branch	Johor Barhu	George	1200000

```
SELECT name
FROM branch
WHERE city = 'Jakarta'
AND assets < 1000000
```

name
Branch_one

SQL DML, simple query example

branch

name	city	director	assets
Clementi	Singapore	Ng Wee Hiong	3000000
Monas	Jakarta	Agus Arianto	4000000

work_for

branch_name	city	employee
Clementi	Singapore	Ng Wee Hiong
Clementi	Singapore	Peter Ho
Clementi	Singapore	Jean Do
Monas	Jakarta	Agus Arianto
Monas	Jakarta	Reza Santi

```
SELECT DISTINCT employee, director
FROM branch, work_for
WHERE name = branch_name
AND branch.city = work_for.city
```

employee	director
Agus Arianto	Agus Arianto
Jean Do	Ng Wee Hiong
Ng Wee Hiong	Ng Wee Hiong
Peter Ho	Ng Wee Hiong
Reza Santi	Agus Arianto

SQL DML, simple query example

```
SELECT DISTINCT work_for.employee, branch.director
FROM branch, work_for
WHERE branch.name = work_for.branch_name
AND branch.city = work_for.city
```

SQL DML, simple query example

```
SELECT DISTINCT employee, director as boss
FROM branch b, work_for w
WHERE name = branch_name
AND b.city = w.city
```

employee	boss
Agus Arianto	Agus Arianto
Jean Do	Ng Wee Hiong
Ng Wee Hiong	Ng Wee Hiong
Peter Ho	Ng Wee Hiong
Reza Santi	Agus Arianto

However... Bag Semantics

- SELECT branch_name FROM work_for

branch_name
Clementi
Clementi
Clementi
Monas
Monas

- SELECT DISTINCT branch_name FROM work_for

branch_name
Clementi
Monas

However... List Semantics

- SELECT name, city
FROM branch
ORDER BY name ASC,
city DESC

SQL Advanced

Arithmetic in SQL

branch	name	city	director	assets
	Clementi	Singapore	Ng Wee Hiong	3000000
	Monas	Jakarta	Agus Arianto	4000000

SELECT name, city, **assets * 1.7** as assets_USD
FROM branch

name	city	assets_USD
Clementi	Singapore	5100000
Monas	Jakarta	6800000

Arithmetic in SQL

- SELECT name, city
FROM branch
WHERE 1700000 > assets * 1.7
- There are numerous other built-in functions for the various data types available

Aggregate Queries

- SELECT **COUNT(DISTINCT *)**
FROM branch

Expr1000
2

Aggregate Queries

name	city	director	assets
Branch_one	Jakarta	Bo Lee	80000
Clementi	Singapore	Ng Wee Hiong	3000000
F_branch	Johor Bahru	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1000000
Monas	Jakarta	Agus Arianto	4000000
S_branch	Johor Bahru	George	1200000

- SELECT **COUNT(DISTINCT city)**
FROM branch

Expr1000
4

Aggregate Queries

name	city	director	assets
Branch_one	Jakarta	Bo Lee	80000
Clementi	Singapore	Ng Wee Hong	3000000
F_branch	Johor Bahru	John	1500000
KL_branch	Kuala Lumpur	Yu Fei	1000000
Moras	Jakarta	Agus Arianto	4000000
S_branch	Johor Bahru	George	1200000

- SELECT COUNT(ALL city)
FROM branch

Expr1000
6

Aggregate Queries

name	city	director	assets
Clementi	Singapore	Ng Wee Hong	3000000
Jakarta_2	Jakarta	George	1000000
Moras	Jakarta	Agus Arianto	4000000
Singapore_2	Singapore	John	1000000

- SELECT AVG(assets)
FROM branch

Expr1000
2250000

Aggregate Queries

branch	name	city	director	assets
	Clementi	Singapore	Ng Wee Hong	3000000
	Jakarta_2	Jakarta	George	1000000
	Moras	Jakarta	Agus Arianto	4000000
	Singapore_2	Singapore	John	1000000

- SELECT city, AVG(assets)
FROM branch
GROUP BY city

city	Expr1001
Jakarta	2500000
Singapore	2000000

Aggregate Queries

- SELECT city, name, AVG(assets)
FROM branch
GROUP BY city, name

This particular query is uninteresting. Why?

- SELECT city, name, assets
FROM branch

It is the same as the above, since (city, name) is a primary key!

Aggregate Queries

- References in the SELECT clause can only be made to aggregates and to attributes in the GROUP BY clause

```
SELECT manager, MIN(assets)
FROM branch
GROUP BY city
```

- is incorrect

Aggregate Queries

```
SELECT manager, MIN(assets)
FROM branch
GROUP BY city, name
```

- is also incorrect! even though it is a key!

- The correct one is:
SELECT manager, MIN(assets)
FROM branch
GROUP BY city, name, manager

Aggregate Queries

- SELECT city
FROM branch
GROUP BY city
WHERE AVG(assets) > 2000000
- Is incorrect!

Aggregate Queries

- SELECT city
FROM branch
GROUP BY city
HAVING AVG(assets) > 2000000

Nested Queries

- SELECT employee
FROM workfor
WHERE branch_name **IN** (
SELECT name
FROM branch
WHERE city= 'Singapore')
- (Note: A red circle highlights the word 'IN' with an arrow pointing to the text '= ANY' written next to it.)*

Nested Queries

- SELECT employee
FROM workfor
WHERE branch_name **NOT IN** (
SELECT name
FROM branch
WHERE city= 'Singapore')
- (Note: A red circle highlights the words 'NOT IN' with an arrow pointing to the text '<> ALL' written next to it.)*

Nested Queries

```
SELECT city  
FROM branch b1  
GROUP BY city  
HAVING AVG(b1.assets) > (  
SELECT AVG(b2.assets)  
FROM branch b2)
```

Nested Queries (Variable Scope)

- A reference to attribute can only be used within the SELECT and WHERE clauses where it is defined or within recursively nested queries

Nested Queries (Variable Scope)

```
SELECT city, AVG(b1.assets), AVG(b2.assets)
FROM branch b1
GROUP BY city
HAVING AVG(b1.assets) > (
  SELECT AVG(b2.assets)
  FROM branch b2)
```

- is incorrect

Nested Query

```
SELECT branch.name, branch.city
FROM branch
WHERE EXISTS (
  SELECT workfor.city
  FROM workfor
  WHERE workfor.employee = 1234
  AND workfor.city = branch.city)
```

Nested Query

- Nested queries sometimes increase the readability of queries
- Nested queries increase the expressive power of SQL

Views

- CREATE VIEW name [schema] AS sql_query

Views

- CREATE VIEW branch_singapore AS
SELECT *
FROM branch
WHERE city = 'Singapore'
- SELECT * FROM branch_singapore

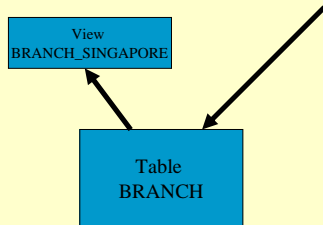
Views

name	city	director	assets
Clement	Kuala Lumpur	Ahmed Abdallah	750000
Clementi	Singapore	Ng Wee Hyong	3000000
East Coast	Singapore	Sanjay Bala	1250000
Jaya	Kuala Lumpur	Putri Ble Ailif	9500000
Lion	Singapore	Kevin Hsu	2500000
Monas	Jakarta	Agus Arianto	900000
Twin Towers	Kuala Lumpur	Alli Muzamed	2000000
Wijaya	Jakarta	Oliver Ooi	1200000

name	city	director	assets
Clementi	Singapore	Ng Wee Hyong	3000000
Lion	Singapore	Kevin Hsu	2500000
East Coast	Singapore	Sanjay Bala	1250000

Views

Update: add, delete, or modify



Views

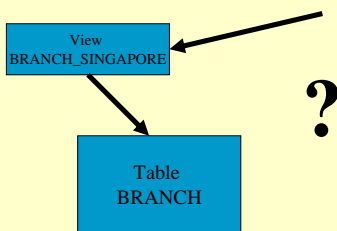
name	city	director	assets
Clemerti	Kuala Lumpur	Ahmed Abdalah	750000
Clemerti	Singapore	Ng Wee Hyong	3000000
East Coast	Singapore	Sanjay Bala	1000000
Jaya	Kuala Lumpur	Putri Ble Ailf	9500000
Lion	Singapore	Kevin Hsu	2500000
Monas	Jakarta	Agus Arianto	900000
Twin Towers	Kuala Lumpur	Alif Mohamed	2000000
Wijaya	Jakarta	Oliver Ooi	1200000

name	city	director	assets
Clemerti	Singapore	Ng Wee Hyong	3000000
Lion	Singapore	Kevin Hsu	2500000
East Coast	Singapore	Sanjay Bala	1000000

Views

- Can one update the views?

Update: add, delete, or modify



Views

- Can one update the views?
- CREATE VIEW branch_singapore AS

```
SELECT *
FROM branch
WHERE city = 'Singapore'
```

Views

name	city	director	assets
Clemerti	Kuala Lumpur	Ahmed Abdalah	750000
Clemerti	Singapore	Ng Wee Hyong	3000000
East Coast	Singapore	Sanjay Bala	1000000
Jaya	Kuala Lumpur	Putri Ble Ailf	9500000
Lion	Singapore	Kevin Hsu	2500000
Monas	Jakarta	Agus Arianto	900000
Twin Towers	Kuala Lumpur	Alif Mohamed	2000000
Wijaya	Jakarta	Oliver Ooi	1200000

name	city	director	assets
Clemens	Singapore	Ng Wee Hyong	3000000
Lion	Singapore	Kevin Hsu	2500000
East Coast	Singapore	Sanjay Bala	1000000

Views

- Can one update the views?
- CREATE VIEW branchname_singapore AS

```
SELECT name
FROM branch
WHERE city = 'Singapore'
```

Views

name	city	director	assets
Clementi	Kuala Lumpur	Ahmed Abdalah	750000
Clementi	Singapore	Ng Wee Hyong	3000000
East Coast	Singapore	Sarjay Bala	1000000
Jaya	Kuala Lumpur	Putri Bte Alif	9500000
Lion	Singapore	Kevin Hsu	2500000
Monas	Jakarta	Agus Arianto	900000
Twin Towers	Kuala Lumpur	Alif Mohamed	2000000
Wijaya	Jakarta	Oliver Ooi	1200000

name
Clementi
Clementi
East Coast
Jaya
Lion
Monas
Twin Towers
Wijaya



Views

- Can one update the views?
- CREATE VIEW assets_singapore AS
SELECT SUM(assets) as total
FROM branch
WHERE city = 'Singapore'

Views

name	city	director	assets
Clementi	Kuala Lumpur	Ahmed Abdalah	750000
Clementi	Singapore	Ng Wee Hyong	3000000
East Coast	Singapore	Sarjay Bala	1000000
Jaya	Kuala Lumpur	Putri Bte Alif	9500000
Lion	Singapore	Kevin Hsu	2500000
Monas	Jakarta	Agus Arianto	900000
Twin Towers	Kuala Lumpur	Alif Mohamed	2000000
Wijaya	Jakarta	Oliver Ooi	1200000

total
6750000



Views

- Logical Data Independence is achieved by means of views
- Views can be pre-compiled
- However views may fool the optimizer