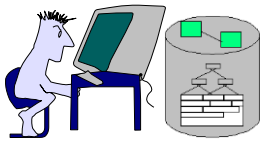
SCS1203 – Part 03

The Relational Model

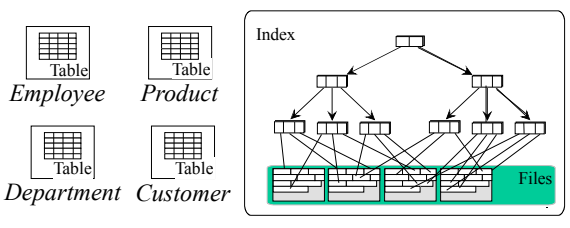


Relational Databases

- Relational DBMS
  - Most common type of DBMS.
  - Data elements are stored in different ***tables*** made up of ***rows*** and ***columns***.
  - ***Relate*** data in different tables through the use of common data element(s).

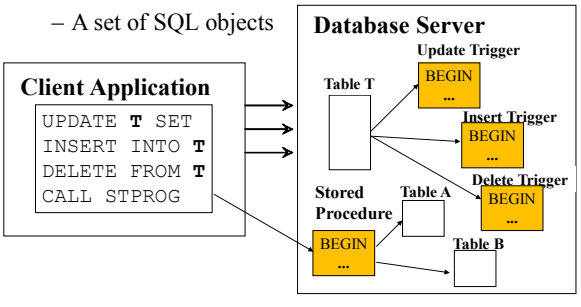
The Relational Objects...

- Database
  - A collection of tables and associated indexes



The Relational Objects...

- Database
  - A set of SQL objects



Stored Procedures are a batch of SQL statements that can be executed in a couple of ways.

Most major DBMs support stored procedures; however, not all do. You will need to verify with your particular DBMS help documentation for specifics.

5

A benefit of stored procedures is that you can centralize data access logic into a single place that is then easy for DBA's to optimize.

Stored procedures also have a security benefit in that you can grant execute rights to a stored procedure but the user will not need to have read/write permissions on the underlying tables.

This is a good first step against SQL Injection.

6

Stored procedures do come with downsides, basically the maintenance associated with your basic CRUD operation.

Let's say for each table you have an *Insert, Update, Delete* and at least one *select* based on the Primary key, that means each table will have 4 procedures. Now take a decent size database of 400 tables, and you have 1600 procedures!
And that's assuming you don't have duplicates which you probably will.

7

The Relational Objects...

- Relation
  - A named, two dimensional table of data
- Database
  - A collection of databases, tables and related objects organised in a structured fashion.
  - *Several database vendors use **schema** interchangeably with **database***

8

Relational Objects...

Data is presented to the user as tables:

- Tables are comprised of ***rows*** and a fixed number of named ***columns***.

Table

	Column 1	Column 2	Column 3	Column 4
Row				
Row				
Row				

Relational Objects...

- ***Column***s are ***attribute***s describing ***an entity***. Each column must have ***a unique name*** and ***a data type***.

Employee

	Name	Designation	Department
Row			
Row			
Row			

Structure of a ***relation*** (e.g. Employee)

Employee(Name, Designation, Department)

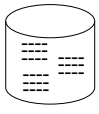
Tables (Relations)

Employee

Name	Designation	Department
De Silva	Manager	Personnel
Perera	Secretary	Personnel
Dias	Manager	Sales

Department

Dept	Manager	Location
Finance	Costa	Colombo
Sales	Alwis	Kandy
Personnel	De Silva	Colombo



Example

Employee

Data Representation

Field or Attribute

Record

Example

Employee file

Name	Designation	Department
De Silva	Manager	Personnel
Perera	Secretary	Personnel
Dias	Salesman	Sales
….	….	

13

Example: **All the data in one file** (simplest)

Employee file

Name, Designation, Department,
Manager, Dept Address, Dept Phone

De Silva	Manager	Personnel	De Silva	Colombo	2589123
Perera	Secretary	Personnel	De Silva	Colombo	2589123
Dias	Salesman	Sales	Alwis	Kandy	2987275
….		….		….	

14

Example: **Data in different files**

Employee file

Name, Designation, Department

De Silva	Manager	Personnel
Perera	Secretary	Personnel
Dias	Salesman	Sales
….	….	….

Department file

Department, Manager,
Dept Address, Dept Phone

Personnel	De Silva	Colombo	589123
Sales	Alwis	Kandy	987275
….	….	….	….

15

Advantages of multiple files

- Can keep data about a Department even if there are no Employees assigned to it
- Entity ***instance***s can exists on its own. i.e. independent of other instances
- Department data are not replicated for all their employees
- Minimise inconsistency problems
  e.g. change of manager

16

- Unique identity of ***an instance*** of an entity.
  - Primary Key
    - employee name?
    - ssn or nid or empno
    - department name
- An attribute can be
  - single valued (e.g. age),
  - multiple value (e.g. office phone no) or
  - composite (e.g. address)

composite key, e.g. (flight_no, date)

Relational Objects

Keys

Primary Key: An attribute (or combination of attributes) that uniquely identifies each row in a relation.

Employee(Emp_No, Emp_Name, Department)

Composite Key: A primary key that consists of more than one attribute

Salary(Emp_No, Eff_Date, Amount)

Relational Objects

Each table has a ***primary key***. The primary key is a column or combination of columns that uniquely identify each row of the table.

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Primary Key

Salary

E-No	Eff-Date	Amt
179	1/1/98	8000
857	3/7/94	9000
179	1/6/97	7000
342	28/1/97	7500

←Primary Key→

Relational Objects

The ***cardinality*** of a table refers to the number of rows in the table. The ***degree*** of a table refers to the number of columns.

Salary Table
Degree = 3
Cardinality = 4

Salary

E-No	Eff-Date	Amt
179	1/1/98	8000
857	3/7/94	9000
179	1/6/97	7000
342	28/1/97	7500

Relationship

works for

Department

Name, Manager

Employee

Name, Designation, Department

Relationship

Employee file

Name, Designation, *Department*

De Silva	Manager	Personnel
Perera	Secretary	Personnel
Dias	Salesman	Sales
….	….	….

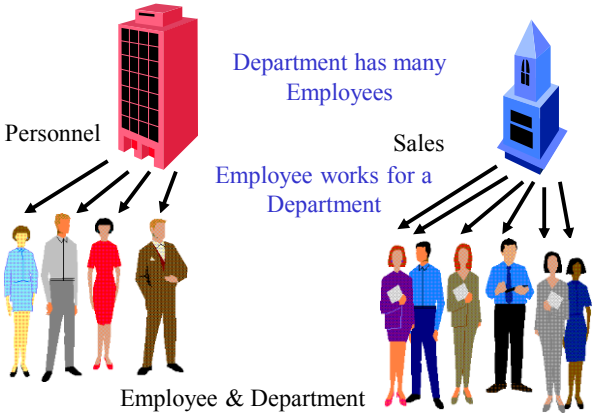
Department file

Department, Manager,

Dept Address, Dept Phone

Personnel	De Silva	Colombo	589123
Sales	Alwis	Kandy	987275
….	….	….	….

A **database** is a group of related files



Employee & Department has one to many relationship

Relational Objects

A *foreign key* is a set of columns in one table that serve as the ***primary key*** in another table

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Primary Key — Foreign Key

Department

D-No	D-Name	M-No
4	Finance	857
7	Sales	179

Primary Key

Recursive foreign key: A foreign key in a relation that references the primary key values of that same relation

Relational Objects...

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Primary Key Foreign Key

Department

D-No	D-Name	M-No
4	Finance	857
7	Sales	179

Primary Key Foreign Key

Salary

E-No	Eff-Date	Amt
179	1/1/98	8000
857	3/7/94	9000
179	1/6/97	7000
342	28/1/97	7500

Foreign Key Primary Key

Rows in one or more tables are associated with each other solely through ***data values*** in columns (no pointers).

Relational Objects

- ***Index***
  - An ordered set of pointers to the data in the table

E-Name	Pointer
De Silva	
Dias	
Perera	
Silva	

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7
719	De Silva	5

Index: Employee Name

E-Name	Pointer
Alwis	
Bandara	
Costa	
De Silva	
Dias	
Opatha	
Peiris	
Perera	
Silva	
Vaas	
Wickrama	
Zoysa	

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7
719	De Silva	5
587	Alwis	4
432	Costa	6
197	Zoysa	2
875	Peiris	4
324	Vaas	7
917	Bandara	3
785	Opatha	2
234	Wickrama	1

Search: Employee Dias

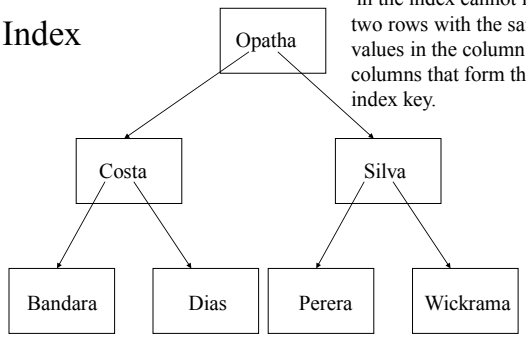
- Index

Improves performance. Access to data is faster

E-Name	Pointer
Alwis	
Bandara	
Costa	
De Silva	
Dias	
Opatha	
Peiris	
Perera	
Silva	
Vaas	
Wickrama	
Zoysa	

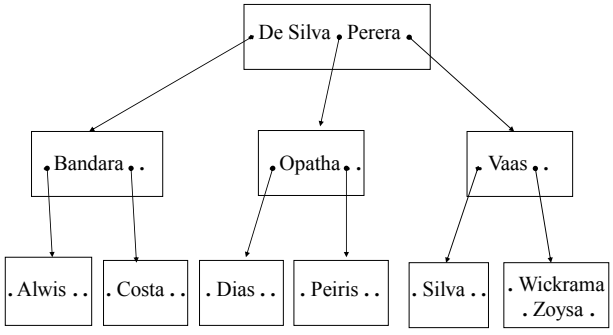
Search: Employee Dias

- Index



Ensures uniqueness. A table with unique fields in the index cannot have two rows with the same values in the column or columns that form the index key.

Search: Employee Dias



Relational Database

STORE
Store Name | City

INVENTORY
Store Name | Part No | Quantity

ORDER
Store Name | Part No | Vendor No | Order No | Quantity

PART
Part No | Description

VENDOR
Vendor No | Vendor Name

STORE
Store 1 | Colombo
Store 2 | Kandy

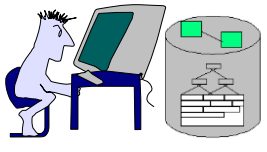
INVENTORY
Store 1 | P1 | 50
Store 1 | P3 | 20
Store 2 | P2 | 100
Store 2 | P1 | 30

ORDER
Store 1 | P3 | 3428 | 0052 | 10
Store 2 | P2 | 3428 | 0098 | 7
Store 2 | P3 | 3428 | 0098 | 15
Store 2 | P4 | 5726 | 0099 | 1

PART
P1 | Printer
P2 | Diskette
P3 | Disk Drive
P4 | Modem

VENDOR
3428 | East West
5726 | DMS

Relational Algebra



Properties Relational Operators

- Relational operations are specified using ***Structured Query Language (SQL)*** *-- a* standard for relational database access.
- Relational operations are ***set level***, meaning that they operate on multiple rows, rather than one record at a time.
- SQL is ***non-procedural***, meaning that the user specifies ***what*** data is to be retrieved rather than ***how*** to retrieve the data.

Properties Relational Operators

- Each operator takes one or more tables as it operand(s) and produces a table as its result.
- Any column value in a table can be referenced, ***not just keys***.
- Operations can be combined to form complex operations.

Relational Operators

Selection **σ :** *horizontal subset of a table*

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

→

Sales Employee

E-No	E-Name	D-No
179	Silva	7
342	Dias	7

Sales-Emp = $\sigma_{\text{D-No=7}}$ (Employee)

Projection **π :** *vertical subset of a table*

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

→

Employee Names

E-No	E-Name
179	Silva
857	Perera
342	Dias

Emp-Names = $\pi_{\text{E-No, E-Name}}$ (Employee)

Cartesian Product ×: *Creates a single table from two tables.*

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Department

D-No	D-Name	M-No
4	Finance	857
7	Sales	179

Emp-Info

E-No	E-Name	D-No	D-No	D-Name	M-No
179	Silva	7	4	Finance	857
857	Perera	4	4	Finance	857
342	Dias	7	4	Finance	857
179	Silva	7	7	Sales	179
857	Perera	4	7	Sales	179
342	Dias	7	7	Sales	179

Emp-Info = Employee × Department

Join ⧖ : *Creates a single table from two tables.*

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Department

D-No	D-Name	M-No
4	Finance	857
7	Sales	179

EquiJoin

Emp-Info

E-No	E-Name	D-No	D-No	D-Name	M-No
179	Silva	7	7	Sales	179
857	Perera	4	4	Finance	857
342	Dias	7	7	Sales	179

Emp-Info = Employee $⧖_{E.D\text{-}No=D.D\text{-}No}$ Department

Joins…

- The most common join is where we only use the '***equal***' operator , and is known as ***equijoin***.
- We can also use other operators (=,<,>,<=, etc…) for the join condition .The ***natural join*** (*) can be used to get rid of the additional attribute in an equijoin condition.
- In a natural join only the matching tuples are displayed. The '***left outer join***' and '***right outer join***' and '***full outer join***'can be used to find even non matching tuples (Refer E&N pp229)

Natural Join *: *Creates a single table from two tables.*

Employee

E-No	E-Name	D-No
179	Silva	7
857	Perera	4
342	Dias	7

Department

D-No	D-Name	M-No
4	Finance	857
7	Sales	179

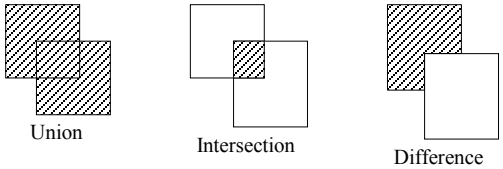
Emp-Info

E-No	E-Name	D-No	D-Name	M-No
179	Silva	7	Sales	179
857	Perera	4	Finance	857
342	Dias	7	Sales	179

Emp-Info = Employee * $_{E.D\text{-}No=D.D\text{-}No}$ Department

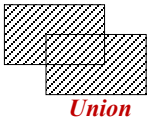
Relational Operators

Other operators



Set operations from mathematical set theory

Set Operators



Union

Student

Fname	Lname
Kapila	Dias
Nimal	Perera
Ajith	Silva
Rohan	Mendis

Instructor

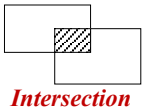
FN	LN
Sunil	De Silva
Kamal	Soysa
Saman	Silva
Kapila	Dias
Nimal	Perera

Stu-Inst

Fname	Lname
Kapila	Dias
Nimal	Perera
Ajith	Silva
Rohan	Mendis
Sunil	De Silva
Kamal	Soysa
Saman	Silva

Stu-Inst = Student ∪ Instructor

Set Operators



Intersection

Student

Fname	Lname
Kapila	Dias
Nimal	Perera
Ajith	Silva
Rohan	Mendis

Instructor

FN	LN
Sunil	De Silva
Kamal	Soysa
Saman	Silva
Kapila	Dias
Nimal	Perera

Stu-Inst

Fname	Lname
Kapila	Dias
Nimal	Perera

Stu-Inst = Student ∩ Instructor

Set Operators

Difference

Student

Fname	Lname
Kapila	Dias
Nimal	Perera
Ajith	Silva
Rohan	Mendis

Instructor

FN	LN
Sunil	De Silva
Kamal	Soysa
Saman	Silva
Kapila	Dias
Nimal	Perera

Stu-Inst

Fname	Lname
Ajith	Silva
Rohan	Mendis

Stu-Inst = Student - Instructor

Inst-Stu = Instructor - Student

Inst-Stu

Fname	Lname
Sunil	De Silva
Kamal	Soysa
Saman	Silva

Complete Set of Relational Algebra Operations

It has been proved that {σ, π, ∪, –, ×} is a complete set.

{Selection, Projection, Union, Difference, Cartesian Product }

Any other relational algebra operator can be expressed in terms of the above operators.

E.g. $R \cap S = (R \cup S) - ((R - S) \cup (S - R))$

Division operator

- Refer Elmasri & Navathe pp 224

Rename operator

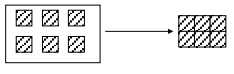
- Refer Elmasri & Navathe pp 215

R(FirstName,LastName,Salary) = $\pi_{Fname,Lname,Sal}$ (Employee)

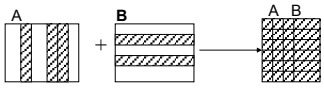
Can be useful for set related operations.

Relational Operators

Because the *result of every relational operation is a table*, operators can be combined to create complex operations. For example:



Select + Project



Project + Select + Join

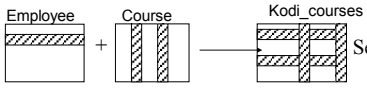
Relational Operators

Get course names thought by lecturer 'Prof Kodikara'
course(cno, cname, lecturer)
employee(empno, ename, designation)

Emp_Kodi ← $\sigma_{eneme='Prof.\ Kodikara'}$ Employee

Courses ← $\pi_{cname,\ lecturer}$ Course

Kodi_courses ← Emp_Kodi * $_{empno\ =\ lecturer}$ Courses



Select + Project + N-Join

Operations on a DBMS

Can be specified using

- Relational Algebra operations (what we learned now)
  - Are usually divided into two groups
    - Set theory operations
    - Operations specifically developed for relational databases
  - But are considered too technical for ordinary users, hence the birth of SQL
  - They are written as a sequence of steps, when executed produce the results
  - Hence the user must give say "how" and not "what" is needed
- Relational calculus
    - Another formal query language which gives 'what' is required, and not how.
    - E.g.:- {t.FNAME,t.LNAME|EMPLOYEE(t) **and** t.SALARY>500}
- SQL

```
SELECT T.FNAME, T.LNAME
FROM EMPLOYEE AS T
WHERE T.SALARY>500
```


END of part 02