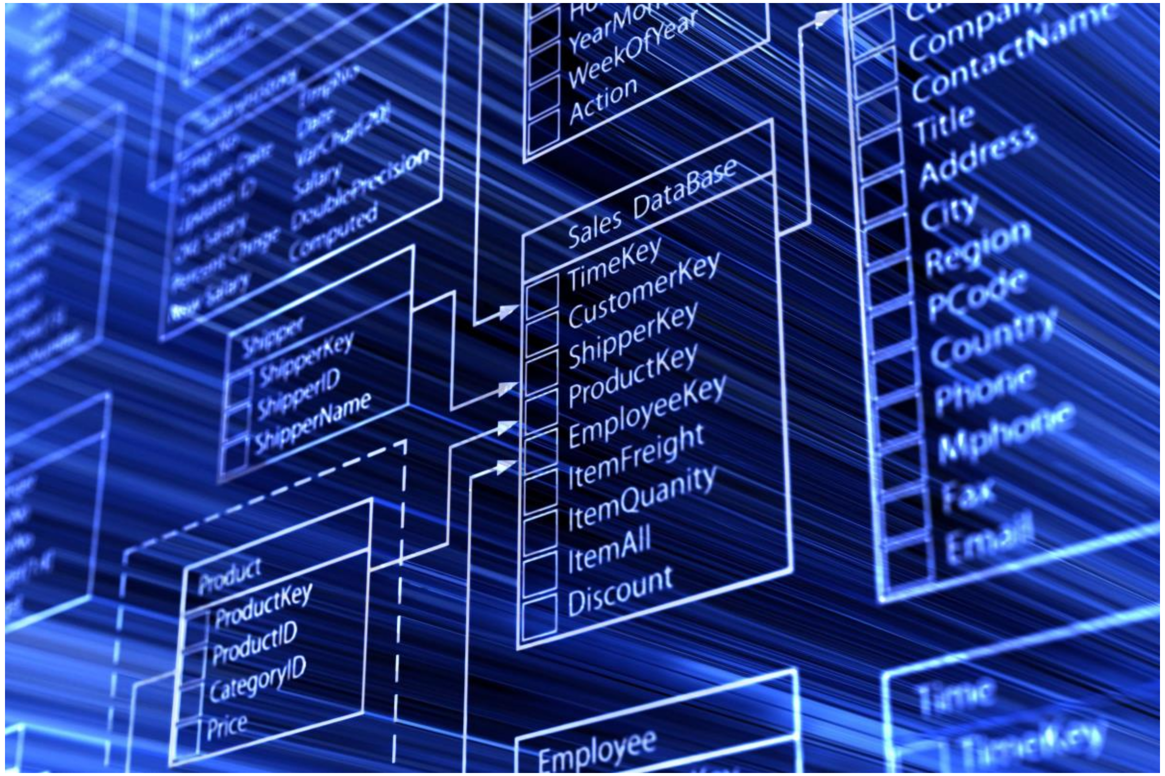

INTRODUCTION TO DATABASES USING ORACLE

420-983-VA

SCHEMA REFINEMENT

SCHEMA REFINEMENT

Additional Features of the E/R Model

Conceptual database design gives us a set of relation schemas and integrity constraints (s) that can be regarded as a good starting point for the final database design. This initial design must be refined by taking the ICs into account more fully than is possible just the E/R model constructs and also by considering performance criteria and typical workloads.

SCHEMA REFINEMENT

Problems Caused by Redundancy

Storing the same information redundantly, that is, in more than one place within a database, can lead to several problems:

- **Redundant Storage**: Some information is stored repeatedly.
- **Update Anomalies**: If one copy of such repeated data is updated, an inconsistency is created unless all copies are similarly updated.
- **Insertion Anomalies**: It may not be possible to store certain information unless some other, unrelated, information is stored as well.
- **Deletion Anomalies**: It may not be possible to delete certain information without losing some other, unrelated, information as well.

SCHEMA REFINEMENT

Problems Caused by Redundancy

Let us consider relation reflecting hourly paid employees:

Hourly_Emps(sin: string, name:string, lot:integer, rating:integer,hourly_wages:float,hours_worked:float)

Suppose that the hourly_wages attribute is determined by the rating attribute. That is for a given rating value, there is only one permissible hourly_wage value.

sin	name	lot	rating	hourly_wage	hours_worked
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

SCHEMA REFINEMENT

Problems Caused by Redundancy

sin	name	lot	rating	hourly_wage	hours_worked
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

- **Redundant Storage**: The rating value 8 corresponds to the hourly_wage 10 and this association is repeated three times.
- **Update Anomalies**: The hourly_wage in the first tuple could be updated without making a similar change in the second tuple.
- **Insertion Anomalie** : We cannot insert a tuple for an employee unless we know the hourly wage for the employee’s rating value.
- **Deletion Anomalie** : If we delete all tuples with a given rating value we lose the association between that rating value and its hourly_wage value.

SCHEMA REFINEMENT

Null Values

Consider the example Hourly_Emps relation. Clearly, null values cannot help eliminate redundant storage or update anomalies. It appears that they can address insertion and deletion anomalies. For instance, to deal with the insertion anomaly, we can insert an employee tuple with null values in the hourly wage field. However, null values cannot address all insertion anomalies. For example, we cannot record the hourly wage for a rating unless there is an employee with that rating, because we cannot store a null value in the sin field, which is a primary key field.

SCHEMA REFINEMENT

Decomposition

A decomposition of a relation schema It consists of replacing the relation schema by two (or more) relation schemas that each contain a subset of the attributes of R and together include all attributes in R. Intuitively, we want to store the information in any given instance of R by storing projections of the instance.

sin	name	lot	rating	hourly_wage	hours_worked
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

sin	name	lot	rating	hours_worked
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

rating	hourly_wage
8	10
5	7

SCHEMA REFINEMENT

Problems Related to Decomposition

Unless we are careful, decomposing a relation schema can create more problems than it solves. Two important questions must be asked repeatedly:

1. Do we need to decompose a relation?
2. What problems (if any) does a given decomposition cause?

To help with the first question, several normal forms have been proposed for relations. If a relation schema is in one of these normal forms, we know that certain kinds of problems cannot arise.

With respect to the second question, two properties of decompositions are of particular interest. The lossless-join property enables us to recover any instance of the decomposed relation from corresponding instances of the smaller relations. The dependency-preservation property enables us to enforce any constraint on the original relation by simply enforcing same constraints on each of the smaller relations. That is, we need not perform joins of the smaller relations to check whether a constraint on the original relation is violated.

From a performance standpoint, queries over the original relation may require us to join the decomposed relations. If such queries are common, the performance penalty of decomposing the relation may not be acceptable. In this case, we may choose to live with some of the problems of redundancy and not decompose the relation.

SCHEMA REFINEMENT

The need for Normalization

To get a better idea of the normalization process, consider the simplified reporting activities of a construction company that manages several building projects. Each project has its own project number, name, assigned employees, and so on. Each employee has an employee number, name, and job classification, such as engineer or computer technician.

The company charges its clients by billing the hours spent on each contract. The hourly billing rate is dependent on the employee's job classification. For example, one hour of computer technician time is billed at a different rate than one hour of engineer time.

SCHEMA REFINEMENT

The need for Normalization

Report that needs to be generated:

PROJECT NUMBER	PROJECT NAME	EMPLOYEE NUMBER	EMPLOYEE NAME	JOB CLASS	CHARGE/ HOUR	HOURS BILLED	TOTAL CHARGE
15	Evergreen	103	June E. Arbough	Elec. Engineer	$ 84.50	23.8	$ 2,011.10
		101	John G. News	Database Designer	$105.00	19.4	$ 2,037.00
		105	Alice K. Johnson *	Database Designer	$105.00	35.7	$ 3,748.50
		106	William Smithfield	Programmer	$ 35.75	12.6	$ 450.45
		102	David H. Senior	Systems Analyst	$ 96.75	23.8	$ 2,302.65
				Subtotal			**$10,549.70**
18	Amber Wave	114	Annelise Jones	Applications Designer	$ 48.10	24.6	$ 1,183.26
		118	James J. Frommer	General Support	$ 18.36	45.3	$ 831.71
		104	Anne K. Ramoras *	Systems Analyst	$ 96.75	32.4	$ 3,134.70
		112	Darlene M. Smithson	DSS Analyst	$ 45.95	44.0	$ 2,021.80
				Subtotal			**$ 7,171.47**
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7	$ 6,793.50
		104	Anne K. Ramoras	Systems Analyst	$96.75	48.4	$ 4,682.70
		113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6	$ 1,135.16
		111	Geoff B. Wabash	Clerical Support	$26.87	22.0	$ 591.14
		106	William Smithfield	Programmer	$35.75	12.8	$ 457.60
				Subtotal			**$13,660.10**
25	Starflight	107	Maria D. Alonzo	Programmer	$ 35.75	24.6	$ 879.45
		115	Travis B. Bawangi	Systems Analyst	$ 96.75	45.8	$ 4,431.15
		101	John G. News *	Database Designer	$105.00	56.3	$ 5,911.50
		114	Annelise Jones	Applications Designer	$ 48.10	33.1	$ 1,592.11
		108	Ralph B. Washington	Systems Analyst	$ 96.75	23.6	$ 2,283.30
		118	James J. Frommer	General Support	$ 18.36	30.5	$ 559.98
		112	Darlene M. Smithson	DSS Analyst	$ 45.95	41.4	$ 1,902.33
				Subtotal			**$17,559.82**
				Total			**$48,941.09**

SCHEMA REFINEMENT

The need for Normalization

We are tasked with creating a database to support this reporting scenario. The first step would be to focus on the base data necessary to generate the report. The total charges, subtotals, and totals are all derived data. Once the initial design is complete, we can make the design decisions about which derived data to store and which to calculate when needed. In this case, the base data is shown below.
The base data is organized around the projects just as the report was organized, with each project having a single row to represent the data associated with that project. The base data shows that a project has multiple employees assigned to it.

Proj_Name	Emp_Number	Emp_Name	Job_Class	Charge_Hour	Hours_Billed
Evergreen	103,101,105,106,102	June E. Arbough,John G. News,Alice K. Johnson*,William Smithfield,David H. Senior	Elec.Engineer,Database Designer,Database Designer,Programmer,Systems Analyst	84.50,105.00,105.00,35.75,96.75	23.8,19.4,35.7,12.6,23.8
Amber Wave	114,118,104,112	Annelise Jones,James J. Frommer,Anne K. Ramoras*,Darlene M. Smithson	Applications Designer,General Support,Systems Analyst,DSS Analyst	48.10,18.36,96.75, 45.95	24.6,45.3,32.40,44
Rolling Tide	105,104,113,111,106	Alice K. Johnson,Anne K. Ramoras,Delbert K. Joenbrood*,Geoff B. Wabash,William Smithfield	Database Designer,Systems Analyst,Applications Designer,Clerical Support,Programmer	105.00,96.75,48.10,26.87,35.75	64.7,$48.40,23.6,22,12.8
Starflight	107,115,101,114,108,000,000	Maria D. Alonzo,Travis B. Bawangi,John G. News*,Annelise Jones,Ralph B. Washington,James J. Frommer,Darlene M. Smithson	Programmer,Systems Analyst,Database Designer,Applications Designer,Systems Analyst,General Support,DSS Analyst	35.75,96.75,105.00,48.10,96.75,18.36,45.95	24.6,45.80,56.3,33.1,23.6,30.5,41.40

SCHEMA REFINEMENT

The need for Normalization

Consider the following deficiencies for the data structure presented before:

1. The data structure invites data inconsistencies. For example, the Job_Class value “Elect. Engineer” might be entered as “Elect.Eng.” in some cases, “El. Eng.” in others, and “EE” in still others. The structure would allow John G. News and Alice K. Johnson in the Evergreen project to charge different rates even though they have the same job classification.

2. The data structure contains several multivalued attributes that make data management tasks very difficult. Because all of the employees working on a project are in a single cell, it is hard to identify each employee individually and for the database to answer questions such as “How many employees are working on the Starlight project?”

3. Employee data is redundant in the table because employees can work on multiple projects. Adding, updating, and deleting data are likely to be very cumbersome using this structure. For example, changing the job classification for Alice K. Johnson would require updating at least two rows.

SCHEMA REFINEMENT

Functional Dependencies

Definition

A functional dependency (FD) on a relation R is a statement of the form “If two tuples of R agree on all of the attributes $A_1,A_2,...,A_n$, then they must also agree on all of another list of attributes $B_1,B_2,...,B_m$. We write this FD formally as $A_1A_2\cdots A_n \rightarrow B_1B_2\cdots B_m$ and say that :

$A_1,A_2,...,A_n$ functionally determine $B_1,B_2,...,B_m$

If we can be sure every instance of a relation R will be one in which a given FD is true, then we say that R satisfies the FD.

SCHEMA REFINEMENT

Functional Dependencies

Example

Let us consider the relation Movies(title, year, length, genre, studioName, starName)

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Do you observe any FDs ?

SCHEMA REFINEMENT

Functional Dependencies

Example

Let us consider the relation Movies(title, year, length, genre, studioName, starName)

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Which of the below are true?

title, year → length, genre, studioName

title, year → starName

genre → studioName

SCHEMA REFINEMENT

Functional Dependencies

Example

Let us consider the relation Movies(title, year, length, genre, studioName, starName)

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Which of the below are true?

title, year → length, genre, studioName ✓

title, year → starName ✗

genre → studioName ✗

SCHEMA REFINEMENT

Functional Dependencies

Keys for Relations

We say a set of one or more attributes $\{A_1,A_2,...,A_n\}$ is a key for a relation R if:

1. Those attributes functionally determine all other attributes of the relation. That is, it is impossible for two distinct tuples of R to agree on all of $A_1,A_2,...,A_n$.

2. No proper subset of $\{A_1,A_2,...,A_n\}$ functionally determines all other attributes of R; i.e., a key must be minimal.

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Can you find a key for Movies relation?

SCHEMA REFINEMENT

Functional Dependencies

Keys for Relations

We say a set of one or more attributes $\{A_1,A_2,...,A_n\}$ is a key for a relation R if:

1. Those attributes functionally determine all other attributes of the relation. That is, it is impossible for two distinct tuples of R to agree on all of $A_1,A_2,...,A_n$.

2. No proper subset of $\{A_1,A_2,...,A_n\}$ functionally determines all other attributes of R; i.e., a key must be minimal.

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Is {title, year, starName} a key?

SCHEMA REFINEMENT

Functional Dependencies

Superkeys/Subkeys

A set of attributes that contains a key is called a superkey, short for “superset of a key.” Thus, every key is a superkey. However, some superkeys are not (minimal) keys. Note that every superkey satisfies the first condition of a key: it functionally determines all other attributes of the relation. However, a superkey need not satisfy the second condition: minimality.

A set of attributes that is a subset of a key is called a subkey.

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

{title, year, startName} is a key , superkey and
{title, year, starName, length, studioName} is a superkey
{title, year} is subkey

SCHEMA REFINEMENT

Functional Dependencies

Reasoning about Functional Dependencies

FD's often can be presented in several different ways, without changing the set of legal instances of the relation. We say:

- Two sets of FD's S and T are equivalent if the set of relation instances satisfying S is exactly the same as the set of relation instances satisfying T.
- More generally, a set of FD's S follows from a set of FD's if every relation instance that satisfies all the FD's in T also satisfies all the FD's in S.

SCHEMA REFINEMENT

Functional Dependencies

Reasoning about Functional Dependencies

Example:
If we are told that a relation R(A,B,C) satisfies the FD's A→B and B→C, then we can deduce that R also satisfies the FD A→C. How does that reasoning go? To prove that A→C, we must consider two tuples of R that agree on A and prove they also agree on C.

Let the tuples agreeing on attribute A be (a, b_1, c_1) and (a, b_2, c_2). Since satisfies A→B, and these tuples agree on A, they must also agree on B. That is, $b_1=b_2$, and the tuples are really (a, b, c_1) and (a, b, c_2), where b is both b_1 and b_2. Similarly, since R satisfies B→C, and the tuples agree on B, they agree on C. Thus, $c_1=c_2$; i.e., the tuples do agree on C. We have proved that any two tuples of R that agree on also agree on C, and that is the FD A→C.

R

A	B	C
a	b_1	c_1
a	b_2	c_2

A→B

B→C

R

A	B	C
a	b	c

$b=b_1=b_2$

$c=c_1=c_2$

A→C

SCHEMA REFINEMENT

Functional Dependencies

The Splitting/Combining Rule

$$A_1A_2A_3\ldots,A_n \rightarrow B_1B_2B_3\ldots,B_m \Leftrightarrow$$

$$A_1A_2A_3\ldots,A_n \rightarrow B_1\ ;$$
$$A_1A_2A_3\ldots,A_n \rightarrow B_2\ ;$$

$$A_1A_2A_3\ldots,A_n \rightarrow B_m\ ;$$

Example:

Employees_Canada(<u>sin</u>, name, zip, str#, city, province)

zip → city, province ; is equivalent to

zip → city

zip → province

SCHEMA REFINEMENT

Functional Dependencies

The Splitting/Combining Rule

Is the following true?

$$A_1A_2A_3\ldots A_n \rightarrow B_1B_2B_3\ldots,B_m \Leftrightarrow$$

$$A_1 \rightarrow B_1B_2B_3\ldots,B_m \;;$$
$$A_2 \rightarrow B_1B_2B_3\ldots,B_m \;;$$

$$A_n \rightarrow B_1B_2B_3\ldots,B_m \;;$$

SCHEMA REFINEMENT

Functional Dependencies

Trivial Functional Dependencies

If $\{B_1,B_2,B_3,\ldots,B_m\} \subseteq \{A_1,A_2,A_3,\ldots,A_n\}$ then $A_1A_2A_3\ldots A_n \rightarrow B_1B_2B_3\ldots B_m$

Example:

Movies(<u>tile</u>, <u>year</u>, length, genre)

length, genre → length, genre
length, genre → length
length, genre → genre

SCHEMA REFINEMENT

Functional Dependencies

The Transitive Rule for FD's

If $A_1A_2A_3\ldots,A_n \rightarrow B_1,B_2,\ldots,B_m$ and $B_1B_2B_3\ldots,B_m \rightarrow C_1,C_2,\ldots,C_k$ then

$$A_1A_2A_3\ldots,A_n \rightarrow C_1,C_2,\ldots,C_k$$

Example:

employee_id → department_id; department_id → department_address

⇒ employee_id → department_address

SCHEMA REFINEMENT

Functional Dependencies

Closure under FD's

The closure of $\{A_1,A_2,\ldots,A_n\}$ under the FD's S is the set of attributes B such that every relation that satisfies all the FD's in set S also satisfies $A_1A_2A_3\ldots,A_n \rightarrow B$. The closure of $\{A_1,A_2,\ldots,A_n\}$ is denoted with $\{A_1,A_2,\ldots,A_n\}^+$. Clearly

$$\{A_1,A_2,\ldots,A_n\}\subseteq\{A_1,A_2,\ldots,A_n\}^+$$

Example:

S={employee_id→employee_name,employee_birthdate;
department_id→department_address;
employee_name,employee_birthdate→department_id;
zip_code→address }

{employee_id}$^+$={employee_id,employee_name,employee_birthdate,department_id,department_address}

If $\{A_1,A_2,\ldots,A_n\}$ is a superkey R, then $\{A_1,A_2,\ldots,A_n\}^+$ contains all attributes in R.

SCHEMA REFINEMENT

Functional Dependencies

Decomposing Relations

The accepted way to eliminate anomalies is to decompose relations. Decomposition of **R** involves splitting the attributes of **R** to make the schemas of two new **S** and **T** relations such that:

1. The union of all attributes from the 2 new relations is the same as the set of attributes in the initial relation.
2. Relations **S** and **T** are the same as **R** projected on the attributes from **S** and **T**.

Example:

Movies(title,year,length,genre,studioName,starName)

Can be decomposed as:

Movies(title,year,length,genre,starName)

MovieStudios(title,length,studioName)

SCHEMA REFINEMENT

Functional Dependencies

Decomposing Relations

The accepted way to eliminate anomalies is to decompose relations. Decomposition of **R** involves splitting the attributes of **R** to make the schemas of two new **S** and **T** relations such that:

1. The union of all attributes from the 2 new relations is the same as the set of attributes in the initial relation.
2. Relations **S** and **T** are the same as **R** projected on the attributes from **S** and **T**.

Example:

Movies(title,year,length,genre,studioName,starName)

Can be decomposed as:

Movies(title,year,length,genre,starName)

MovieStudios(title,length,studioName)

SCHEMA REFINEMENT

Functional Dependencies

Elementary FD's

Let G be a set of FDs. A functional dependency $X \rightarrow A$ from G is said to be elementary wrt. G, when $A \notin X$ and G^+ does not contain a FD $X' \rightarrow A$, with $X' \subset X$.

An elementary FD for a relation R is one which is elementary wrt. FDs of R.

Example:

Movies(title, year, genre, studioName)

title,year → year - not elementary for Movies
title, year, genre → studioName - not elementary for Movies
title, year → studioName - elementary for Movies

SCHEMA REFINEMENT

Functional Dependencies

Minimal Basis of a set of FD's

Let F be a set of FDs. A set G of functional dependencies is say to be the minimal basis for F if following holds.

1. F is equivalent with G
2. Each FD in G have singleton right sides. That is they have only one attribute on the right hand of the dependency.
3. Each FD in G are elementary (see: Elementary definition).
4. By removing any FD from G the resulted set will not be equivalent with F.

Example:

$F=\{A \rightarrow B;\ A,B \rightarrow C,D\}$ the minimal basis for F is $G=\{A \rightarrow B;\ A \rightarrow C;\ A \rightarrow D\}$

SCHEMA REFINEMENT

Functional Dependencies

Prime attributes

An attribute A from a relation is called prime attribute if A is part of a key in R. The rest of attributes from R are called non-prime attributes.

Example:

Gradings(studentId, subjectCode, subjectName, exam#, score, grade)

What are the candidate keys for Gradings?

SCHEMA REFINEMENT

Functional Dependencies

Prime attributes

An attribute A from a relation is called prime attribute if A is part of a key in R. The rest of attributes from R are called non-prime attributes.

Example:

Gradings(studentId, subjectCode, subjectName, exam#, score, grade)

{stundentId, subjectCode, exam#}

{stundentId, subjectName exam#}

Prime attributes {stundentId, subjectCode, subjectName, exam#},
Non-Prime attributes {grade, score}