ជំពូកទី ២

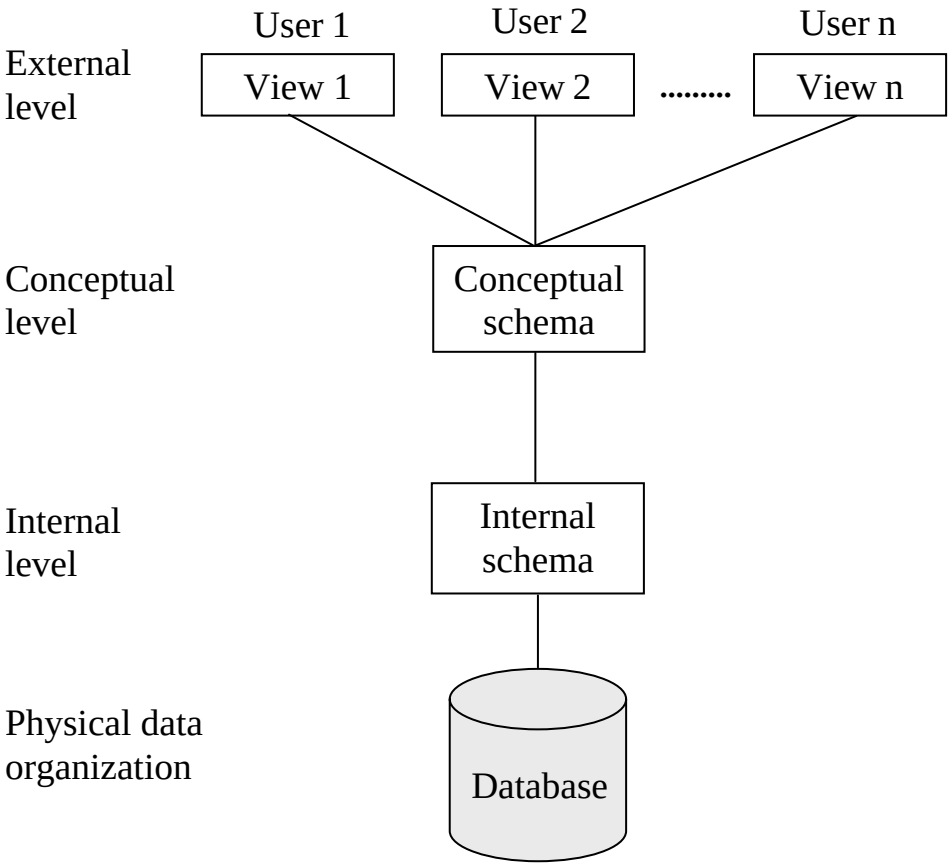
Database Environment

គោលបំណងជាចំបងរបស់ database system គឺផ្តល់អោយ users នូវ abstract view of data ដោយលាក់បាំងព័ត៌មានលំអិតថាតើត្រូវរក្សាទុក និង manipulate ទិន្នន័យដោយរបៀបណា។ ហេតុដូចនេះហើយ ចំនុចចាប់ផ្តើមក្នុងការកសាង database ធ្វើយ៉ាងណាអោយ abstract and general description នៃតំរូវការព័ត៌មានរបស់ organization បង្ហាញក្នុង database។ ម្យ៉ាងវិញទៀត ដោយសារតែ database is a shared resource ហេតុដូចនេះ users ច្រើនគ្នាអាចធ្វើការបង្ហាញទិន្នន័យក្នុងទំរង់ផ្សេងៗគ្នាចំពោះទិន្នន័យដូចគ្នាដែលរក្សាទុកក្នុង database ដើម្បីបំពេញទៅនឹងតំរូវការនេះ architecture of most commercial DBMSs ប្រើប្រាស់សព្វថ្ងៃមានមូលដ្ឋានលើ ANSI-SPARC architecture។

1. The Three-Level ANSI-SPARC Architecture

នៅក្នុងឆ្នាំ 1971 DBTG (Data Base Task Group) តែងតាំងដោយ Conference on Data Systems and Languages (CODASYL) បានលើកសំណើទៅលើ standard terminology and general architecture for database system ដោយបានបែងចែកជា two-level approach គឺ system view (វីអាចហៅថា **schema**) និង user view (វីអាចហៅថា **subschemas**)។ Terminology and architecture ស្រដៀងគ្នាមួយទៀតដែលលើកសំណើក្នុងឆ្នាំ 1975 ដោយ American National Standard Institute (ANSI) Standards Planning and Requirements Committee (SPARC) គឺ ANSI/X3/SPARC។ ANSI-SPARC ត្រូវបានទទួលស្គាល់ចំពោះតំរូវការ three-level approach ជាមួយ system catalog។ ទោះបីជា ANSI-SPARC model មិនទាន់ក្លាយជា standard នៅឡើយក៏វានៅតែផ្តល់នូវមូលដ្ឋានគ្រឹះក្នុងការស្វែងយល់មួយចំនួនចំពោះ functionality នៃ DBMS។

Three-level architectures រួមមាន **external**, **conceptual** និង **internal** level ដូចបង្ហាញក្នុងរូប 2.1 ខាងក្រោម។ គោលដៅរបស់ three-level architecture ដើម្បីបំបែកការយល់ឃើញរបស់អ្នកប្រើប្រាស់លើ database និងរបៀបដែលវារក្សាទុកនៅលើផ្នែ disk (to separate each user's view of the database from the way it is physically represented)។



(រូប 2.1)

ហេតុផលមួយចំនួនក្នុងការបបែកអោយដាច់ពីគ្នានេះគឺ:

- User **នីមួយៗគួរតែអាចប្រើប្រាស់ទិន្នន័យដូចគ្នា ប៉ុន្តែមានការបង្ហាញទិន្នន័យអោយឃើញខុសៗពីគ្នា** (different customized view of data)។ User **នីមួយៗគួរតែអាចកែប្រែរបៀបដែលគាត់មើល** (view) **ទិន្នន័យ ហើយការកែប្រែនេះមិនបានធ្វើអោយប៉ះពាល់ដល់** users **ផ្សេងទៀតទេ**។

- User **មិនគួរអាចធ្វើការដោយផ្ទាល់ជាមួយ** physical database storage details **ដូចជា** indexing **ជាដើម**។

- Database Administrator (DBA) **គួរតែអាចកែប្រែ** database storage structures **ដោយមិនធ្វើអោយប៉ះពាល់ដល់** users' views។

- រចនាសម្ព័ន្ធខាងក្នុងនៃ database (internal structure of database) គួរតែនៅរក្សាដដែល ទោះបីជាមានការកែប្រែលើទ្រង់ទ្រាយខាងក្រៅ (physical aspect) នៃ storage ដូចជាការផ្លាស់ប្តូរទៅកាន់ new storage device ក៏ដោយ។

- DBA គួរតែអាចកែប្រែ conceptual or global structure នៃ database ដោយគ្មានផលប៉ះពាល់ដល់ users ទាំងអស់ទេ។

Users មើលឃើញទិន្នន័យតាមរយៈ **external level** ហើយ DBMS និង operating system មើលឃើញទិន្នន័យតាមរយៈ **internal level**។ Internal level គឺជាកន្លែងដែលទិន្នន័យរក្សាទុកដោយប្រើប្រាស់ data structure និង file organizations។ **Conceptual level** ផ្តល់នូវការបំលែង (mapping) និង independence រវាង external and internal levels។

1. 1. External Level

External level The users' view of the database. This level describes that part of the database that is relevant to each user.

External level សំដៅទៅលើការមើលឃើញរបស់ users ទៅលើ database។ Level នេះបង្ហាញអំពីផ្នែកនៃ database ដែលពាក់ព័ន្ធជាមួយ user នីមួយៗ។ External level ផ្តុំកនូវការមើលឃើញទៅលើ database ផ្សេងៗគ្នាជាច្រើន (a number of different external views of the database) ហើយ user នីមួយៗមាន 'real world view' បង្ហាញក្នុងទំរង់មួយដែល user អាចយល់បាន ដោយធ្វើការបញ្ចូលតែ entities, attributes និង relationships ណាដែលពាក់ព័ន្ធទៅនឹង view នោះតែប៉ុណ្ណោះ។ Entities, attributes, or relationships ដែលមិនបានពាក់ព័ន្ធ ប្រហែលជាបង្ហាញក្នុង database ប៉ុន្តែ user មិនបានដឹងអំពីបញ្ហានេះទេ។

លើសពីនេះទៀត ការមើលឃើញផ្សេងៗគ្នា (different views) អាចមានការបង្ហាញផ្សេងៗគ្នាចំពោះទិន្នន័យតែមួយ។ ឧទាហរណ៍: User ប្រហែលជាមើល dates ក្នុងទំរង់ (day, month, year) ខណៈពេលនោះមាន users ផ្សេងទៀតអាចមើល dates ក្នុងទំរង់ជា (year, month, day)។ Users មួយចំនួនប្រហែលជាបញ្ចូល derived or calculated data ដែលជាទិន្នន័យមិនបានរក្សាទុកក្នុង database ប៉ុន្តែបង្កើតវាឡើងនៅពេលដែលត្រូវការ។ ឧទាហរណ៍: យើងប្រហែលចង់មើល age នៃបុគ្គលិក ប៉ុន្តែយើងមិនបានរក្សាទុក age attribute ទៅកាន់ database ទេព្រោះវាទាមទារអោយធ្វើការកែប្រែរៀងរាល់ឆ្នាំ ដូចនេះយើងប្រហែលជារក្សា date

of birth វិញ ព្រោះវាមិនចាំបាច់ធ្វើការកែប្រែជារៀងរាល់ឆ្នាំទេ ហើយម្យ៉ាងទៀតយើងអាចគណនារក age ឃើញដោយប្រើប្រាស់អនុគមន៍ទៅលើ date of birth។

1. 2. Conceptual Level

Conceptual level The community view of the database. This level describes *what* data is stored in the database and the relationship among the data.

Conceptual level ពណ៌នាថាតើទិន្នន័យអ្វីខ្លះត្រូវរក្សានៅក្នុង database និង relationship ចំពោះទិន្នន័យនោះ។ The middle level នៅក្នុង three-level architecture គឺ conceptual level ហើយ level នេះផ្តុក logical structure នៃ database ទាំងមូល ដែលមើលឃើញដោយ DBA។ វាគឺជាការមើលឃើញពេញលេញ (complete view) នៃតំរូវការទិន្នន័យរបស់ organization ដែលមានលក្ខណៈ independence ទៅលើ storage considerations។ Conceptual level បង្ហាញនូវ:

- all entities, their attributes, and their relationships;
- the constraints on the data;
- security and integrity information.

Conceptual level ត្រូវតែមិនផ្តុក storage-dependent details។ ឧទាហរណ៍: conceptual level ផ្តុកនូវ description of entity ដោយផ្តុកតែ data types of attributes (ឧទាហរណ៍: integer, real, character) និង length (ដូចជា maximum numbers of digits or characters) ប៉ុន្តែមិនបានផ្តុក storage considerations ណាមួយឡើយដូចជា number of bytes occupied ជាដើម។

1. 3. Internal Level

Internal level The physical representation of the database on the computer. This level describes *how* the data is stored in the database.

Internal level ពណ៌នាថាតើត្រូវរក្សាទុកទិន្នន័យទៅក្នុង database ដោយរបៀបណា។ Internal level សំដៅទៅលើ physical implementation of the database ដើម្បីសំរេច optimal runtime performance and storage space utilization ជាពិសេសលើ data structures and file organizations សំរាប់រក្សាទិន្នន័យលើ storage devices។ វារួមជាមួយ operating system access method (file management techniques for storing and retrieving records) ដើម្បីផ្តុកទិន្នន័យនៅលើ storage devices, បង្កើត indexes, ការទទួលទិន្នន័យ.... ។ Internal level ពាក់ព័ន្ធជាមួយបញ្ហាមួយចំនួនដូចជា:

- storage space allocation for data and indexes;
- record descriptions for storage (with stored sizes for data items);
- record placement;
- data compression and data encryption techniques.

នៅពីក្រោម internal level គឺជា **physical level** ដែលប្រហែលជាត្រូវគ្រប់គ្រងដោយ operating system ក្រោមការដឹកនាំរបស់ DBMS ទោះបីជាយ៉ាងណាក៏ដោយ functions of DBMS and operating system នៅ physical level នៅមិនទាន់កំនត់អោយបានដាច់ស្រឡះនៅឡើយទេ។ DBMSs មួយចំនួនទាញយកផលប្រយោជន៍ពី operating system access methods ជាច្រើន នៅពេលដែល DBMSs ផ្សេងទៀតប្រើប្រាស់តែ most basic access methods តែប៉ុណ្ណោះ។

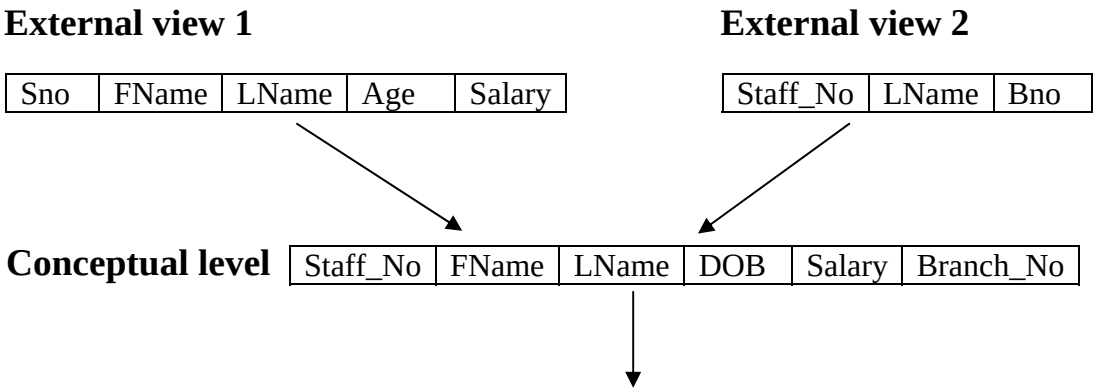
1. 4. Schemas, Mapping, and Instances

ការពិពណ៌នាទៅលើ database ទាំងមូលគេហៅថា database schema (The overall description of the database is called the **database schema**.)។ ដោយយោងទៅលើ three-level architecture មាន 3 កំរិតដូចនេះ schema ក៏ចែកជា 3 ប្រភេទផងដែរគឺ ចំពោះកំរិតខ្ពស់ជាងគេ មាន **external schemas** (រឺ **subschemas**) ជាច្រើនដែលត្រូវគ្នាជាមួយការមើលឃើញទិន្នន័យផ្សេងៗគ្នា (different views of data) ចំពោះ conceptual level គេហៅថា **conceptual schema** និងកំរិតក្រោមបំផុតគេហៅថា **internal schema**។

Conceptual schema ពិពណ៌នាគ្រប់ data items និង relationship between data items ព្រមទាំង integrity constraints ទាំងអស់ ហើយនៅក្នុង database មួយមាន conceptual schema តែមួយគត់។ Internal schema គឺជាការពិពណ៌នាទាំងស្រុងនៃ internal model ដោយផ្តុំកន្លែង definitions of stored records, the methods of representation the data fields and the indexes ហើយមានតែមួយគត់គ្រប់ database។

DBMS ទទួលបន្ទុកក្នុងការបំលែង (mapping) schemas ទាំង 3 ប្រភេទនេះ ហើយវាក៏ត្រួតពិនិត្យទៅលើ schemas for inconsistency ផងដែរ ម្យ៉ាងវិញទៀត DBMS ត្រូវតែត្រួតពិនិត្យរាល់ external schema ទាំងអស់ដែលកើតពី conceptual schema និងប្រើប្រាស់ព័ត៌មាននៅក្នុង conceptual schema ដើម្បីបំលែង (map) រវាង external schema និង internal schema នីមួយៗអោយស្គាល់គ្នា។ Conceptual schema ទាក់ទងជាមួយ internal schema តាមរយៈ **conceptual/internal mapping** ហើយ conceptual/internal mapping ធ្វើអោយ DBMS មានលទ្ធភាពស្វែងរកទិន្នន័យពិត (actual data) រឺបណ្តុំ records

នៅក្នុង physical storage សំរាប់បង្កបង្កើតជា logical record នៅក្នុង conceptual schema និងការប្រតិបត្តិ constraints ណាមួយលើ logical record (any constraints to be enforced on the operations for that logical record)។ វាជួយដោះស្រាយផងដែរ ចំពោះភាពខុសគ្នានៃ entity names, attribute names, attribute order, data types, and so on។ ជាចុងក្រោយ វាល់ external schema នីមួយៗទាក់ទងជាមួយ conceptual schema តាមរយៈ **external/conceptual mapping** ហើយ external/conceptual mapping ធ្វើអោយ DBMS មានលទ្ធភាពបំលែងឈ្មោះដែលមើលឃើញដោយ user ទៅកាន់ផ្នែកពាក់ព័ន្ធ នឹង conceptual schema (map names in the user's view onto the relevant part of the conceptual schema)។



(រូប 2.2)

តាមរយៈរូប 2.2 យើងសង្កេតឃើញថា មានការមើលឃើញ 2 ផ្សេងពីគ្នាចំពោះព័ត៌មានលំអិតបុគ្គលិក (two different views of staff details) ទីមួយផ្តុំកតែ a staff number, a first and last name, an age, and a salary ហើយទីពីផ្តុំកតែ a staff number, a last name, and branch number។ External views ទាំងនេះត្រូវបានគេច្រចាច់បញ្ចូលគ្នាទៅជា conceptual view តែមួយ ហើយនៅក្នុងដំណើរការនេះ ភាពខុសគ្នាជាចំបងគឺថា age attribute បានបំលែងជា date of birth, DOB attribute។ DBMS ថែរក្សា external/conceptual

mapping ឧទាហរណ៍: ដូចជាបំលែង Sno attribute នៃ first external view ទៅជា Staff_No attribute ក្នុង conceptual level ហើយ conceptual level ត្រូវបំលែងទៅកាន់ internal level ដោយផ្ទុកតែ physical description of the structure in a high-level language។ គួរកត់សំគាល់ផងដែរថា លំដាប់ទីតាំង attributes នៅក្នុង internal level ខុសពី conceptual level ហើយ DBMS មានតួនាទីក្នុងការបំលែងវា (លំដាប់ attributes)។

វាមានសារៈសំខាន់ណាស់ក្នុងការបែងចែកអោយដាច់ស្រឡះរវាង description of database និង database។ Description of database is the **database schema** ដែល schema នេះត្រូវបានគេកំនត់កំឡុងពេលដំណើរការកសាង database ហើយកំរនឹងធ្វើការកែប្រែជាញឹកញាប់ណាស់។ ទោះបីជាយ៉ាងណា ទិន្នន័យពិតប្រាកដ (actual data) នៅក្នុង database កែប្រែជារឿយៗ ឧទាហរណ៍: វាកែប្រែរៀងរាល់ពេលដែលយើងបញ្ចូលព័ត៌មានលំអិតបុគ្គលិកថ្មី រឺ property ថ្មី។ ទិន្នន័យនៅក្នុង database ក្នុងកំឡុងពេលណាមួយគេហៅថា **database instance** ហេតុដូចនេះ មាន database instances ជាច្រើនចំពោះ database schema តែមួយ។ Schema ជួនកាលគេហៅថា **intension** of the database ហើយ instance គេហៅថា **extension** (or **state**) of the database។

1. 5. Data Independence

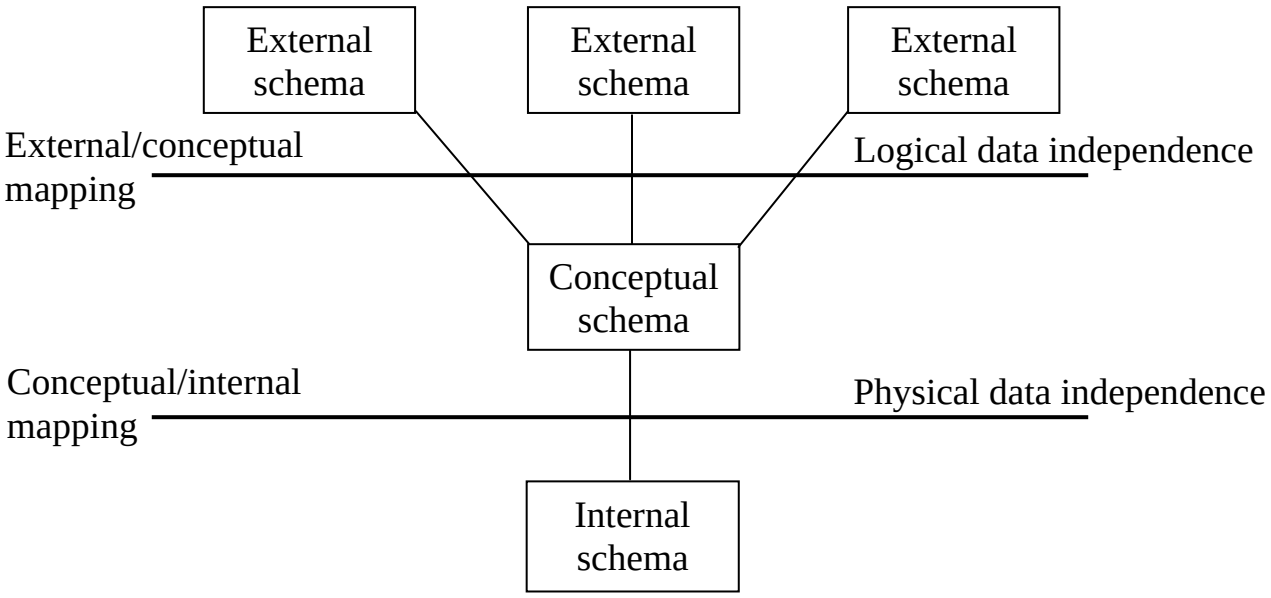
គោលបំណងជាចំបងរបស់ three-level architecture គឺផ្តល់នូវ **data independence** មានន័យថា កំរិតខាងលើគ្មានការប៉ះពាល់នៅពេលកែប្រែចំពោះកំរិតក្រោម (means that upper levels are unaffected by changes to lower levels)។ Data independence ចែកជា 2 ប្រភេទគឺ logical and physical។

Logical data independence Logical data independence refers to the immunity of external schemas to changes in the conceptual schema.

Logical data independence សំដៅលើការកែប្រែ conceptual schema មិនបានធ្វើអោយប៉ះពាល់ដល់ external schemas។ ការកែប្រែ conceptual schema ដូចជា បន្ថែម រឺលុប entities, attributes, or relationships គួរតែអាចធ្វើទៅបានដោយមិនធ្វើអោយ external schemas កែប្រែតាម រឺ application programs សរសេរសារជាថ្មី។

Physical data independence Physical data independence refers to the immunity of the conceptual schema to changes in the internal schema.

Physical data independence សំដៅលើការកែប្រែ internal schema មិនបានធ្វើអោយប៉ះពាល់ដល់ conceptual schema។ ការកែប្រែ internal schema ដូចជាប្រើប្រាស់ different file organizations or storage structures, different storage devices, modifying indexes គួរតែអាចធ្វើទៅបានដោយមិនធ្វើអោយ conceptual schema រឺ external schemas កែប្រែតាម។ ចំពោះទស្សនៈរបស់ users ការកែប្រែ internal schema វាបង្កអោយមានផលប៉ះពាល់ចំពោះ performance។ រូប 2.3 ខាងក្រោមបង្ហាញពី data independence នីមួយៗកើតឡើងនៅក្នុងទំនាក់ទនងជាមួយ three-level architecture:



(រូប 2.3)

2. Database Languages

Data sublanguage ចែកជា 2 ផ្នែកគឺ **Data Definition Language** (DDL) និង **Data Manipulation Language** (DML)។ DDL ប្រើប្រាស់សំរាប់បង្កើត database schema ហើយ DML ប្រើប្រាស់សំរាប់អាន (read) និងកែប្រែ (update) database។ ភាសាទាំងនេះគេហៅថា **data sublanguage** ព្រោះវាមិនបានបញ្ចូល constructs for all computing needs ដូចពួក high-level programming languages ទេ។ DBMS ជាច្រើនមាន facility សំរាប់បញ្ចូល (embedded) data sublanguage ទៅក្នុង high-level programming languages ដូចជា COBOL, Fortran, Pascal, C, or Visual Basic។ ក្នុងករណីនេះ high-level programming language ជួនកាលគេអាចហៅថា **host language**។ ដើម្បី compile the embedded file ជាដំបូង វាលុបរាល់ commands ទាំងអស់ក្នុង data sublanguage ចេញពី host-language program ហើយជំនួសវិញដោយ

function calls។ បន្ទាប់មក compile pre-processed file ហើយដាក់ក្នុង object module, linked with a library ផ្នុក replaces functions ផ្តល់ដោយ DBMS ហើយដំណើរការនៅពេលត្រូវការ។ Data sublanguages ភាគច្រើនផ្តល់ជាំងដែរនូវលក្ខណៈ non-embedded or **interactive** ដោយអនុញ្ញាតិអោយ commands បញ្ចូលដោយផ្ទាល់ពី terminal។

2. 1. The Data Definition Language (DDL)

DDL A descriptive language that allows the DBA or user to describe and name the entities required for the application and the relationships that may exist between the different entities.

DDL គឺជា descriptive language ដែលអនុញ្ញាតិអោយ DBA រឺ user រៀបរាប់និងដាក់ឈ្មោះ entities ត្រូវការសំរាប់ application និង relationships ដែលកើតមានឡើងចំពោះ entities ទាំងនោះ។ DDL ប្រើប្រាស់សំរាប់បង្កើត រឺកែប្រែ database schema ប៉ុន្តែមិនអាចប្រើសំរាប់ manipulate data បានទេ។ លទ្ធផលនៃ compilation of the DDL statements គឺជាសំនុំ tables រក្សាទុកក្នុង files ពិសេស គេហៅថា **system catalog**។ System catalog ផ្នុក **meta-data** (data describes objects in the database) និងធ្វើអោយវាមានភាពងាយស្រួលក្នុងការចូលប្រើប្រាស់ (access or manipulate)។ Meta-data ផ្នុក definitions of records, data items, and other objects ដែល user ចាប់អារម្មណ៍រឺត្រូវការដោយ DBMS។ DBMS ជាទូទៅធ្វើការពិគ្រោះយោបល់ជាមួយ system catalog មុននឹងប្រើប្រាស់ទិន្នន័យពិតប្រាកដក្នុង database។

3. 2. The Data Manipulation Language (DML)

DML A language that provides a set of operations to support the basic data manipulation operations on the data held in the database.

DML គឺជាភាសាមួយដែលផ្តល់នូវបណ្តុំ operations ដើម្បីទ្រទ្រង់ basic data manipulation operations ទៅលើទិន្នន័យរក្សាទុកក្នុង database។ Data manipulation operations តែងតែមានលក្ខណៈខាងក្រោម:

- the insertion of new data into the database,
- the modification of data stored in the database,
- the retrieval of data contained in the database,
- the deletion of data from the database.

Main function មួយក្នុងចំណោម functions ទាំងឡាយផ្តល់ដោយ DBMS គឺទ្រទ្រង់ data manipulation language។ DML ចែកជា 2 ប្រភេទដោយយោលទៅលើភាពខុសប្លែកគ្នា

ក្នុងការទទួលយកទិន្នន័យគឺ **procedural** and **non-procedural**។ ជាទូទៅ procedural language ធ្វើការជាមួយ records ម្តងមួយៗ តែ non-procedural language ធ្វើការជាមួយ សំនុំ records។

Procedural DMLs

Procedural DML A language that allows the user to tell system what data is needed and exactly *how* to receive the data.

Procedural DML គឺជាភាសាមួយដែលអនុញ្ញាតិអោយ user ប្រាប់ប្រព័ន្ធនូវទិន្នន័យអ្វីខ្លះ ដែលត្រូវការ និងតើត្រូវទទួលទិន្នន័យដោយរបៀបណា។ មានន័យថា ជាមួយ procedural language, user រឺក៏ programmer ត្រូវតែកំនត់ទិន្នន័យណាខ្លះដែលត្រូវការ និងវិធីក្នុងការទទួលទិន្នន័យនោះ គឺថា user ត្រូវបញ្ជេញរាល់ data access operations ទាំងអស់ដែលត្រូវការដោយធ្វើការ calling appropriate procedures ដើម្បីទទួលបានព័ត៌មានត្រូវការ។ Procedural DML ទទួលបាន a record ហើយប្រតិបត្តិការលើវាដើម្បីទទួលបានលទ្ធផល និងទទួល record បន្តបន្ទាប់ទៀតម្តងមួយៗ រហូតទាល់តែវាទទួលបានទិន្នន័យត្រូវការទាំងអស់។ Network and hierarchical DMLs ជាទូទៅមានលក្ខណៈ procedural។

Non-procedural DMLs

Non-procedural DML A language that allows the user to state *what* data is needed than *how* it is to be retrieved.

Non-procedural DML គឺជាភាសាមួយដែលអនុញ្ញាតិអោយ user បញ្ជាក់នូវទិន្នន័យអ្វីដែលត្រូវការជាជាងវិធីដើម្បីទទួលបានទិន្នន័យទាំងនោះ។ DBMS បកប្រែពី DML statements ទៅជា a procedure (a set of procedures) ដែល manipulate សំនុំទិន្នន័យដែលវាត្រូវការ។ នេះវាបានធ្វើអោយ user មិនចាំបាច់ដឹងពី data structures are internally implemented and algorithms ដែលត្រូវការដើម្បីទទួលទិន្នន័យ និងបំលែងទិន្នន័យ។ Non-procedural language គេអាចហៅថា *declarative language*។ Relational DBMSs ជាច្រើនបានបញ្ចូលទំរង់មួយចំនួននៃ non-procedural language for data manipulation ដូចជា SQL (Structured Query Language) or QBE (Query-By-Example)។ Non-procedural DMLs ជាទូទៅងាយស្រួលក្នុងការសិក្សាជាង procedural DMLs ហើយ user ធ្វើការតិចជាង DBMS។

2. 3. 4GL

4GL ជាតួអក្សរកាត់ចំពោះ **Fourth-Generation Language**។ 4GL is essentially a shorthand programming language ដែលបំលែង operations នៅក្នុង third-generation language រាប់រយជួរអោយនៅសល់តែ 10 ទី 20 ជួរតែប៉ុណ្ណោះក្នុង 4GL។

4GL គឺជា non-procedural ដែលទាមទារអោយ user កំនត់ទិន្នន័យអ្វីខ្លះដែលត្រូវការ ខុសពី 3GL មានលក្ខណៈជា procedural។ 4GL ពឹងពាក់ភាគច្រើនទៅលើ higher-level components គេហៅថា fourth-generation tools។ User មិនចាំបាច់កំនត់ជំហានដែល program ត្រូវការដើម្បីប្រតិបត្តិកិច្ចការទេ ផ្ទុយទៅវិញដោយគ្រាន់តែបំពេញ parameters សំរាប់ tools នោះហើយប្រើវាសំរាប់បង្កើត application program។ SQL and QBE ជាឧទាហរណ៍ នៃ 4GLs។

3. Data Models

យើងដឹងហើយថា គេបង្កើត schema ដោយប្រើប្រាស់ data definition language។ តាមពិតទៅ គេអាចសរសេរ data definition language នោះសំរាប់ DBMS ណាមួយ ជាក់លាក់តែម្តង ប៉ុន្តែគួរអោយសោកស្តាយដោយសារតែ language នោះវា too low-level សំរាប់ពិពណ៌នាពីតំរូវការទិន្នន័យនៃ organization ក្នុងទំរង់មួយដែលអាចអោយ users ជាច្រើនប្រភេទ អាចមើលយល់បាន ដូចនេះវាទាមទារអោយមាន high-level description of the schema គឺ **data model**។

Data model An integrated collection of concepts for describing data, relationships between data, and constraints on the data in an organization.

Data model គឺបណ្តុំ concepts ជាច្រើនសំរាប់ពិពណ៌នាទិន្នន័យ, relationship រវាងទិន្នន័យ និង constraints លើទិន្នន័យនៃ organization។ Model គឺជាការបង្ហាញចំពោះ ‘real world’ objects and events, and their associations។ វាផ្តល់នូវ basic concepts and notations ដែលអនុញ្ញាតិអោយអ្នកកសាង database (database designer) និង end-user យល់យ៉ាងត្រឹមត្រូវ និងច្បាស់លាស់នូវ organization data។ Data model ផ្ទុកនូវ components ចំនួន 3 គឺ:

1. A structural part ផ្ទុកនូវសំនុំ rules កំនត់ទៅលើ database។

2. A manipulative part កំនត់ប្រភេទ operations ដែលអនុញ្ញាតិអោយធ្វើការទៅលើទិន្នន័យ (operations មានដូចជា updating or retrieving data and changing the structure of the database)។

3. សំនុំ integrity rules ប្រើប្រាស់សំរាប់ធ្វើអោយទិន្នន័យមានភាពត្រឹមត្រូវ។

គោលបំណងរបស់ data model សំរាប់បង្ហាញទិន្នន័យ និងធ្វើអោយទិន្នន័យនោះមើលងាយយល់ ដូច្នេះធ្វើអោយការកសាង database មានលក្ខណៈងាយស្រួល។ Data model ចែកចេញជា:

3. 1. Relational Data Model

Relational data model មានមូលដ្ឋានគ្រឹះលើទស្សនៈនៃ mathematical relations។ នៅក្នុង relational model ទិន្នន័យ និងទំនាក់ទំនង (relationships) បង្ហាញជា tables ដែល table នីមួយៗមានបណ្តុំនៃ columns ដែលមានឈ្មោះខុសៗគ្នា។ នៅក្នុងរូប 2.4 បង្ហាញពីព័ត៌មាននៃបុគ្គលិក (staff) និងសាខា (branch) ឧទាហរណ៍: វាបានបង្ហាញអំពីបុគ្គលិកឈ្មោះ Ny Tony ស្នាក់នៅផ្ទះលេខ 19 Sothearos St, PP មានតួនាទីជា manager ទទួលបានប្រាក់ខែ 30000 ធ្វើការនៅសាខា B5 មានអាស័យដ្ឋាននៅ 22 Chak Anre St, PP។ គួរធ្វើការកត់សំគាល់ដែរថា មានទំនាក់ទំនងរវាង staff និង branch: សមាជិកនៃបុគ្គលិកធ្វើការនៅតាមសាខា។ ចំពោះ relational data model ទាមទារអោយអ្នកប្រើប្រាស់យល់ វិគិតថា រាល់ទិន្នន័យនៅក្នុង database រក្សាក្នុងទំរង់ជា tables។

Staff_No	SName	SAddress	Position	Salary	Branch_No
S21	Ny Tony	19 Sothearos St, PP	Manager	30000	B5
S37	Ann Ford	34 Monivong St, PP	Snr Asst	12000	B3
S14	Khun Vitou	234 Tonele Mekong St, PP	Deputy	18000	B3
S9	Phal Socheat	13 Tonele Mekong St, PP	Assistant	9000	B7
S5	Pak Vannly	17 Sihanouk St, PP	Manager	24000	B3
S41	Thay Sokan	37 Sihanouk St, PP	Assistant	9000	B5

Staff relation

Branch_No	BAddress	Tel_No
B3	163 Chba Ampov St, PP	023 210003
B5	22 Chak Anre St, PP	023 217272
B7	72 Chak Anre St, PP	023 725456

Branch relation (រូប 2.4)

3. 2. Network Data Model

នៅក្នុង network data model ទិន្នន័យបង្ហាញក្នុងទំរង់ជាបណ្តុំនៃ records និង relationships បង្ហាញជា sets។ ផ្ទុយពី relational model, relationships បង្ហាញជា sets ដែលក្លាយជា pointers នៅពេលប្រតិបត្តិ។

3. 3. Hierarchical Data Model

Hierarchical model គឺជាការដាក់កំរិតបន្ថែមទៀតទៅលើ network data model មានន័យថា នៅក្នុង hierarchical data model ទិន្នន័យបង្ហាញក្នុងទំរង់ជាបណ្តុំនៃ records និង relationships បង្ហាញជា sets ប៉ុន្តែវាអនុញ្ញាតិអោយ node មួយមាន parent តែមួយគត់។

4. Functions of a DBMS

Codd បានលើកឡើងពីសេវាទាំង 10 ដែល DBMS គួរតែផ្តល់:

(1) Data storage, retrieval, and update

DBMS ត្រូវតែផ្តល់អោយ users នូវលទ្ធភាពក្នុងការរក្សាទុក, ទទួល និងកែប្រែទិន្នន័យក្នុង database ហើយវាគឺជា function មូលដ្ឋានគ្រឹះ។

(2) A user-accessible catalog

DBMS ត្រូវតែផ្តល់ catalog ដែលពិពណ៌នា data items រក្សាទុក និងតើ data items ណាខ្លះដែល users អាចប្រើប្រាស់បាន។ លក្ខណៈចំបងនៃ ANSI-SPARC architecture គឺការស្គាល់នូវ integrated system catalog ក្នុងការរក្សាទុកទិន្នន័យអំពី schemas, users, applications, and so on។ System catalog or data dictionary គឺជាកន្លែងផ្ទុកព័ត៌មានរៀបរាប់ពីទិន្នន័យនៅក្នុង database មានន័យថា ‘data about data’ or **meta-data**។ ជាទូទៅ system catalog រក្សា:

- Names, types, and sizes of data items.
- Names of relationships.
- Integrity constraints on the data.
- Names of authorized users who have access to the data.
- External, conceptual, and internal schemas and the mapping between the schemas.
- Usage statistics, such as the frequencies of transactions and counts on the number of accesses made to objects in the database.

(3) Transaction support

DMBS ត្រូវតែផ្តល់នូវ mechanism ដើម្បីធ្វើអោយប្រាកដថា វាបានដំណើរការរាល់ការកែប្រែទិន្នន័យនៅក្នុង transaction រឺក៏មិនបានដំណើរការទាល់តែសោះ។ Transaction

គឺជាសំណុំនៃសកម្មភាពដែលដំណើរការដោយ a single user or application program ចូលទៅប្រើប្រាស់ រឺកែប្រែមាតិការនៃ database។ ឧទាហរណ៍: transaction ធម្មតាមួយ ដែលធ្វើការបញ្ចូលទិន្នន័យទៅកាន់ staff table, transaction ធ្វើការកែប្រែប្រាក់ខែ សមាជិកណាមួយជាដើម។

(4) Concurrency control services

DBMS ត្រូវតែផ្តល់នូវ mechanism ដើម្បីធ្វើអោយប្រាកដថា ការកែប្រែទិន្នន័យនៅក្នុង database ប្រព្រឹត្តទៅដោយត្រឹមត្រូវ នៅពេលដែលអ្នកប្រើប្រាស់ជាច្រើនកែប្រែទិន្នន័យក្នុងពេលដំណាលគ្នា។

(5) Recovery services

DBMS ត្រូវតែផ្តល់នូវ mechanism សំរាប់ recover database ប្រសិនបើ database មានបញ្ហាដោយប្រការណាមួយ ដូចជា system crash, media failure, hardware or software error... ។

(6) Authorization services

DBMS ត្រូវតែផ្តល់នូវ mechanism ដើម្បីធ្វើអោយប្រាកដថា មានតែអ្នកប្រើប្រាស់ដែលគេផ្តល់សិទ្ធ (authorized users) តែប៉ុណ្ណោះអាចចូលទៅប្រើប្រាស់ database។

(7) Support for data communication

DBMS ត្រូវតែមានលទ្ធភាពច្របាច់ចូលគ្នាជាមួយ communication software។ Users ភាគច្រើនចូលទៅប្រើប្រាស់ database តាមរយៈ dumb terminals ដែលភ្ជាប់ដោយផ្ទាល់ទៅកាន់កុំព្យូទ័រផ្ទុក DBMS។ ក្នុងករណីនេះ DBMS ទទួលសំណើពី user តាមការបញ្ជូនលើបណ្តាញ network ហើយដំណើរការសំណើនោះដោយបញ្ជូនលទ្ធផលទៅកាន់ dumb terminals វិញ។

(8) Integrity services

DBMS ត្រូវតែផ្តល់នូវមធ្យោបាយដើម្បីធានាថា ទាំងទិន្នន័យនៅក្នុង database និងការកែប្រែទិន្នន័យគោរពទៅតាម rules ដែលគេបានកំនត់។ Database integrity សំដៅទៅលើភាពត្រឹមត្រូវ និងភាពស៊ីសង្វាក់គ្នានៃទិន្នន័យរក្សាទុកក្នុង database ហើយជាទូទៅ integrity បង្ហាញក្នុងទំរង់ជា constraints ឧទាហរណ៍: យើងធ្វើការកំនត់ constraint ទៅលើ sex field ដោយអោយបញ្ចូលតែតួអក្សរ ‘M’ រឺ ‘F’ តែប៉ុណ្ណោះ។

(9) Services to promote data independence

DBMS ត្រូវតែទ្រទ្រង់ភាពឯករាជ្យ (independence) នៃ programs ពីរចនាសម្ព័ន្ធពិតប្រាកដនៃ database។

(10) Utility services

DBMS គួរតែផ្តល់នូវ utilities មួយចំនួនដូចជា:

- Import and export facilities.
- Monitoring facilities to monitor database usage and operation.
- Statistical analysis programs
- Index reorganization facilities
- Garbage collection and reallocation

Review Questions

1. ចូរពន្យល់អោយបានក្បោះក្បាយពី data independence, DML, DDL?
2. ចូរពិពណ៌នាកំរិតទាំង 3 នៃ ANSI-SPARC architecture?
3. ចូរពន្យល់ពី data model?
4. ចូររៀបរាប់ពី function of a DBMS អោយបានក្បោះក្បាយ?