LECTURE NOTE ON

CMP 327: DATA MANAGEMENT

300 L

Volume I

COMPUTER SCIENCE/INFORMATION TECHNOLOGY
BINGHAM UNIVERSITY, KARU

BY

DR.SANI F.A

Data Management

Data management is the practice of collecting, keeping, and using data securely, efficiently, and cost-effectively. The goal of data management is to help people, organizations, and connected things optimize the use of data within the bounds of policy and regulation so that they can make decisions and take actions that maximize the benefit to the organization. A robust data management strategy is becoming more important than ever as organizations increasingly rely on data to create value.

Databases

Efficient data management can be vital to success of any business. Even something as simple as a customer mailing list needs to be managed appropriately if you want it to remain up to date and accurate. Tools or applications such as databases can make these tasks easier and more efficient.

The database is one of the cornerstones of information technology, it has capacity to organize, process and manage information in a structured and controlled manner.

Definitions

- A database is a comprehensive collection of data organized for easy access.
- It is a collection of logically related records.
- It is a repository for data and the engine for managing access to data.

Database use in business

You can use business databases to help organize and manage your customers, your business inventory and employees. Databases can streamline your:

- customer management processes
- inventory tracking
- employee database
- productivity reporting
- financial reporting
- data analysis
- Supply chain management

The secret to the successful use of database tools and technology is the way in which information is structured

Motivation for implementation of databases

The implementation of databases was prompted by the following problems of file systems environment:

- Lack of data integration
- Lack of standard
- Lack of central control
- Inconsistency
- Redundancy
- Lack of multiple view

Advantages of database

- Reduced data redundancy
- Reduced updating errors and increased consistency
- Greater data integrity and independence from applications programs
- Improved data access to users through use of host and query languages
- Improved data security
- Reduced data entry, storage, and retrieval costs
- Facilitated development of new applications program
- Centralized control of data
- Data can be viewed in multiple ways
- Standard can be easily enforced.

Disadvantages

- Database systems are complex, difficult, and time-consuming to design
- Substantial hardware and software start-up costs
- Damage to database affects virtually all applications programs
- Extensive conversion costs in moving form a file-based system to a database system
- Initial training required for all programmers and users

Database Management System (DBMS)

Database management system is a software which is used to manage the database. For example: MySQL, Oracle, SQL Server, IBM DB2, PostgreSQL, Amazon SimpleDB (cloud based) , etc are a very popular commercial database which used in different applications.

DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.

It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

DBMS allows users the following tasks:

1. **Data Definition:** It is used for creation, modification, and removal database and database objects.

2. **Data Manipulation:** It is used for the insertion, modification, and deletion of the actual data in the database.

3. **Data Retrieval:** It is used to retrieve the data from the database which can be used by applications for various purposes.

4. **User Administration:** It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by unexpected failure.

Characteristics of Database Management System

A database management system has following characteristics:

1. **Data stored into Tables:** Data is never directly stored into the database. Data is stored into tables, created inside the database. DBMS also allows to have relationships between tables which makes the data more meaningful and connected. You can easily understand what type of data is stored where by looking at all the tables created in a database.
2. **Reduced Redundancy:** In the modern world hard drives are very cheap, but earlier when hard drives were too expensive, unnecessary repetition of data in

database was a big problem. But DBMS follows **Normalization** which divides the data in such a way that repetition is minimum.

3. **Data Consistency:** On Live data, i.e. data that is being continuously updated and added, maintaining the consistency of data can become a challenge. But DBMS handles it all by itself.
4. **Support Multiple user and Concurrent Access:** DBMS allows multiple users to work on it(update, insert, delete data) at the same time and still manages to maintain the data consistency.
5. **Query Language:** DBMS provides users with a simple Query language, using which data can be easily fetched, inserted, deleted and updated in a database.
6. **Security:** The DBMS also takes care of the security of data, protecting the data from un-authorised access. In a typical DBMS, we can create user accounts with different access permissions, using which we can easily secure our data by restricting user access.
7. DBMS supports **transactions**, which allows us to better handle and manage data integrity in real world applications where multi-threading is extensively used.

Advantages of DBMS

- Segregation of application program.
- Minimal data duplication or data redundancy.
- Easy retrieval of data using the Query Language.
- Reduced development time and maintenance need.
- With Cloud Datacenters, we now have Database Management Systems capable of storing almost infinite data.
- Seamless integration into the application programming languages which makes it very easier to add a database to almost any application or website.

Disadvantages of DBMS

- It's Complexity
- Except MySQL, which is open source, licensed DBMSs are generally costly.
- They are large in size.

Functions of a DBMS

So, what does a DBMS really do? It organizes your files to give you more control over your data.

A DBMS makes it possible for users to create, edit and update data in database files. Once created, the DBMS makes it possible to store and retrieve data from those database files.

More specifically, a DBMS provides the following functions:

Concurrency: concurrent access (meaning 'at the same time') to the same database by multiple users

Security: security rules to determine access rights of users

Backup and recovery: processes to back-up the data regularly and recover data if a problem occurs

Integrity: database structure and rules improve the integrity of the data

Data descriptions: a data dictionary provides a description of the data

Database Administration

Administering the database is the responsibility of the Database Administrator (DBA). The role of the DBA includes:

- Selecting the server hardware on which the database software will run
- Installing and configuring the Oracle 10*g* software on the server hardware
- Creating the database itself
- Creating and managing the tables and other objects used to manage the application data

- Creating and managing database users
- Establishing reliable backup and recovery processes for the database
- Monitoring and tuning database performance

DATA ABSTRACTION

The major purpose of a database system is to provide users with an abstract view of the system. The system hides certain details of how data is stored and maintained.

Complexity also should be hidden from database users. There are several levels of abstraction:

(a) **Internal Level**

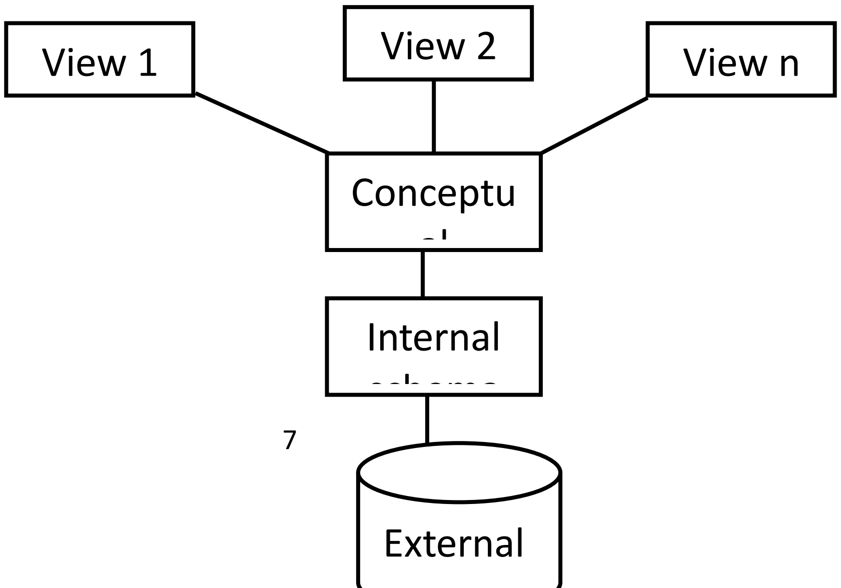
- Shows the physical representation of database on the computer, how the data are stored. e.g. index, B-tree, hashing. Record descriptions for storage, record placement and data compress and data encryption

Conceptual Level

- Defines logical structure of a database, describes all entities, their attributes, and their relationship. Describes the constraints on the data, semantic information about the data and security and integrity information

(c) **External View Level**

- Describes part of the database for a particular group of users, how users see the data. e.g. tellers in a bank get a view of customer accounts, but not of payroll data.



Database Models

A **database model** is a type of data model that determines the logical structure of a database and fundamentally determines in which manner data can be stored, organized and manipulated.

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed.

There are many kinds of data models. Some of the most common ones include:

Hierarchical database model

Relational model

Network model

Object-oriented database model

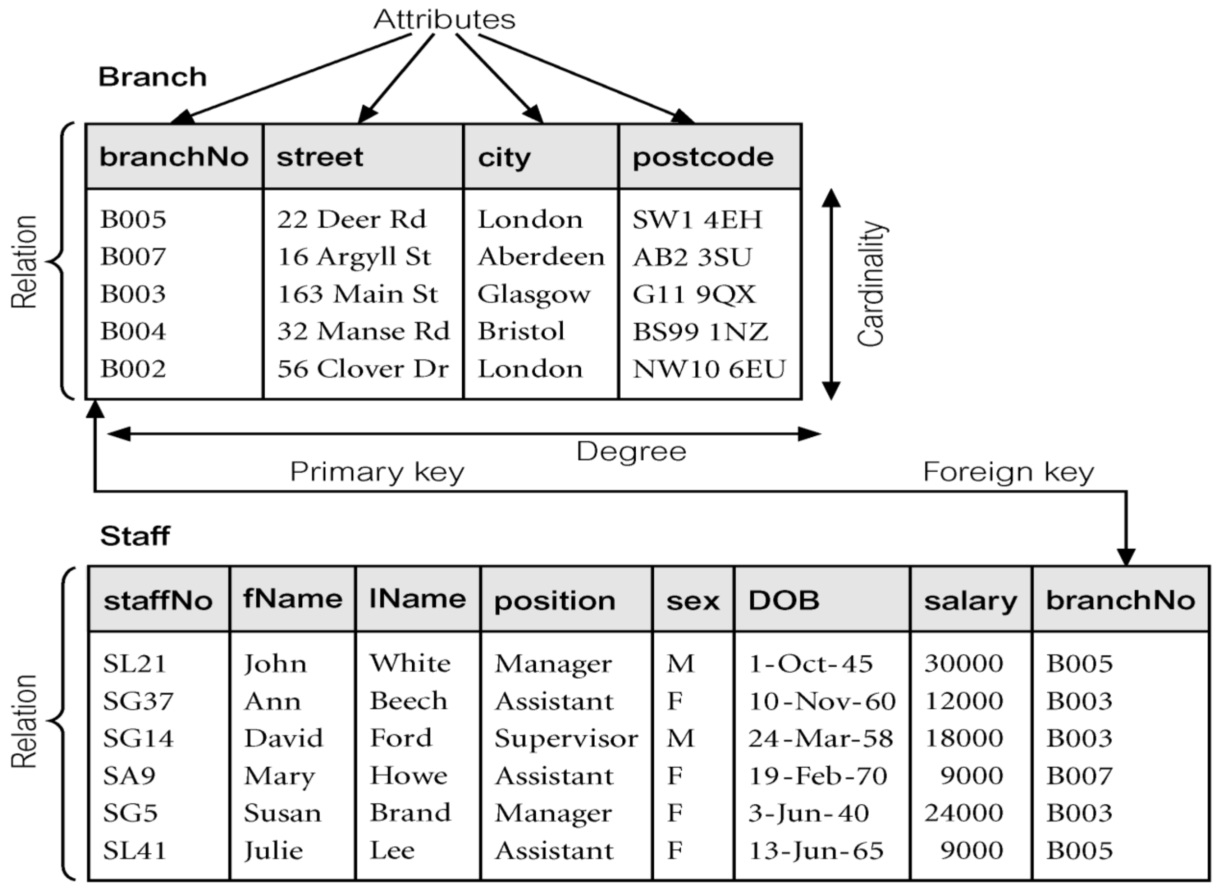
The most popular database model is the relational database model which was introduced by E. F Codd in 1970. Most databases in use such as Oracle, MySQL, and SQL SERVER are relational databases.

Relational Model Terminologies

- A ***relation*** is a table with columns and rows.
    - Only applies to logical structure of the database, not the physical structure.
    - A relation corresponds to an entity set, or collection of entities.
- ***An entity*** is a person, place, event, or thing about which data is collected.
- ***Attribute*** is a named column of a relation.
    - It corresponds to a characteristic of an entity.
    - They are also called fields
- ***Tuple*** is a row of a relation.
- ***Degree*** is the number of attributes in a relation.

- ***Cardinality*** is the number of tuples in a relation
- ***Domain*** is the set of allowable values for one or more attributes.
- ***Relational Database*** is a collection of normalized relations with distinct relation names.
- ***Relation schema*** is named relation defined by a set of attribute and domain name pairs.

Relational database schema *is a s*et of relation schemas, each with a distinct name



Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Properties of Relations

- Relation name is distinct from all other relation names in relational schema.
- Each cell of relation contains exactly one atomic (single) value.
- Each attribute has a distinct name.
- Values of an attribute are all from the same domain.
- Each tuple is distinct; there are no duplicate tuples.
- Order of attributes has no significance.
- Order of tuples has no significance, theoretically.

Data Modeling

Data modeling is the process of creating a data model for the data to be stored in a Database. This data model is a conceptual representation of

- Data objects
- The associations between different data objects
- The rules.

A Data Model is used to document, define, organize, and show how the data structures within a given database, architecture, application, or platform are connected, stored, accessed, and processed within the given system and between other systems.

Data modeling helps in the visual representation of data and enforces business rules, regulatory compliances, and government policies on the data. Data Models ensure consistency in naming conventions, default values, semantics, and security while ensuring quality of the data.

Data model emphasizes on what data is needed and how it should be organized instead of what operations need to be performed on the data. Data Model is like architect's building plan which helps to build a conceptual model and set the relationship between data items.

Uses Data Model

The primary goal of using data model are:

- Ensures that all data objects required by the database are accurately represented. Omission of data will lead to creation of faulty reports and produce incorrect results.
- A data model helps design the database at the conceptual, physical and logical levels.
- Data Model structure helps to define the relational tables, primary and foreign keys and stored procedures.
- It provides a clear picture of the base data and can be used by database developers to create a physical database.
- It is also helpful to identify missing and redundant data.
- Though the initial creation of data model is labor and time consuming, in the long run, it makes your IT infrastructure upgrade and maintenance cheaper and faster.

Types of Data Models

There are mainly three different types of data models:

Conceptual: This Data Model defines **WHAT** the system contains. This model is typically created by Business stakeholders and Data Architects. The purpose is to organize, scope and define business concepts and rules.

1. **Logical:** Defines **HOW** the system should be implemented regardless of the DBMS. This model is typically created by Data Architects and Business Analysts. The purpose is to developed technical map of rules and data structures.
2. **Physical**: This Data Model describes **HOW** the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database.

Conceptual Model

The main aim of this model is to establish the entities, their attributes, and their relationships. In this Data modeling level, there is hardly any detail available of the actual Database structure.

The 3 basic elements of the conceptual model are Entity, Attribute, Relationship.

ER MODEL – BASIC CONCEPTS

The ER model defines the conceptual view of a database. It works around real-world entities and the associations among them. At view level, the ER model is considered a good option for designing databases.

Entity

An entity can be a real-world object, either animate or inanimate, that can be easily identifiable. For example, in a school database, students, teachers, classes, and courses offered can be considered as entities. All these entities have some attributes or properties that give them their identity.

An entity set is a collection of similar types of entities. An entity set may contain entities with attribute sharing similar values. For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties. Entity sets need not be disjoint.

Attributes

Entities are represented by means of their properties called **attributes**. All attributes have values. For example, a student entity may have name, class, and age as attributes.

There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

Types of Attributes

- **Simple attribute:** Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.

- **Composite attribute:** Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first_name and last_name.

- **Derived attribute:** Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For example, average_salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data_of_birth.

- **Single-value attribute:** Single-value attributes contain single value. For example: Social_Security_Number.

- **Multi-value attribute:** Multi-value attributes may contain more than one values. For example, a person can have more than one phone number, email_address, etc.

Set and Keys

Key is an attribute or collection of attributes that uniquely identifies an entity among entity set. For example, the roll_number of a student makes him/her identifiable among students.

- ➢ **Super Key**: A set of attributes (one or more) that collectively identifies an entity in an entity set.
- ➢ **Candidate Key**: A minimal super key is called a candidate key. An entity set may have more than one candidate key.
- ➢ **Primary Key**: A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

Relationship

The association among entities is called a relationship. For example, an employee **works_at** a department, a student **enrolls** in a course. Here, Works_at and Enrolls are called relationships.

Relationship Set

A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called **descriptive attributes**.

Degree of Relationship

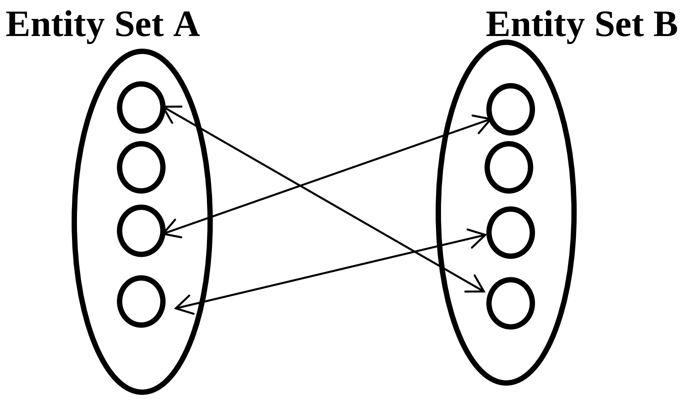
The number of participating entities in a relationship defines the degree of the relationship.

- Binary = degree 2
- Ternary = degree 3
- n-ary = degree

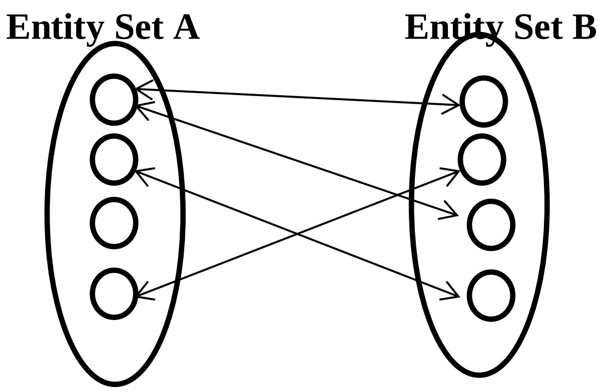
Mapping Cardinalities

Cardinality defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.

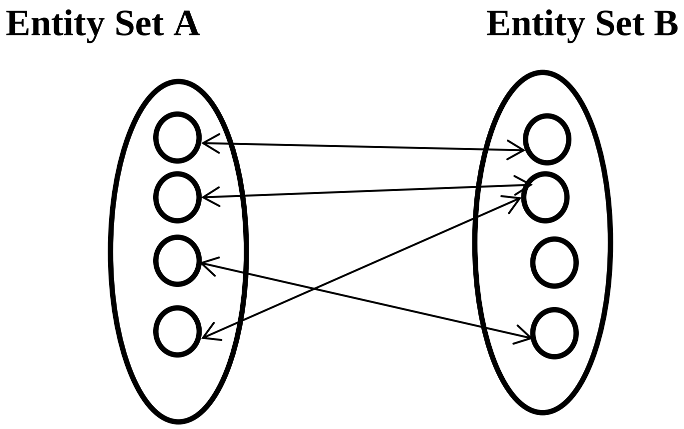
- **One-to-one**: One entity from entity set A can be associated with at most one entity of entity set B and vice versa.



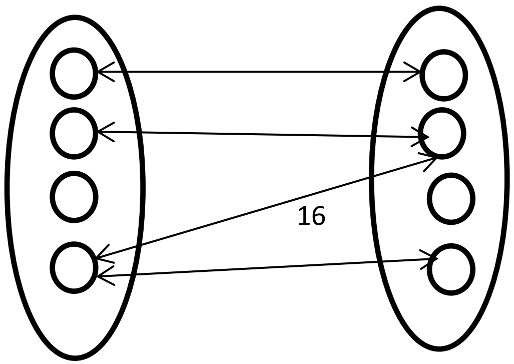
- **One-to-many**: One entity from entity set A can be associated with more than one entities of entity set B, however an entity from entity set B can be associated with at most one entity.



❖ **Many-to-one**: More than one entities from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A.



❖ **Many-to-many**: One entity from A can be associated with more than one entity from B and vice versa.



Characteristics of a conceptual data model

- Offers Organization-wide coverage of the business concepts.
- This type of Data Models are designed and developed for a business audience.
- The conceptual model is developed independently of hardware specifications like data storage capacity, location or software specifications like DBMS vendor and technology. The focus is to represent data as a user will see it in the "real world."

Logical Data Model

Logical data models add further information to the conceptual model elements. It defines the structure of the data elements and set the relationships between them.

Firstname (string)
Age (number)

The advantage of the Logical data model is to provide a foundation to form the base for the Physical model. However, the modeling structure remains generic.

At this Data Modeling level, no primary or secondary key is defined. At this Data modeling level, you need to verify and adjust the connector details that were set earlier for relationships.

Characteristics of a Logical data model

- Describes data needs for a single project but could integrate with other logical data models based on the scope of the project.
- Designed and developed independently from the DBMS.
- Data attributes will have data types with exact precisions and length.
- Normalization processes to the model is applied typically till 3NF.

Physical Data Model

A Physical Data Model describes the database specific implementation of the data model. It offers an abstraction of the database and helps generate schema. This is because of the richness of meta-data offered by a Physical Data Model.

Firstname (Varchar)
Age (integer)
Empno (Varchar) Primary key

This type of Data model also helps to visualize database structure. It helps to model database columns keys, constraints, indexes, triggers, and other RDBMS features.

Characteristics of a physical data model:

- The physical data model describes data need for a single project or application though it may be integrated with other physical data models based on project scope.
- Data Model contains relationships between tables that which addresses cardinality and nullability of the relationships.
- Developed for a specific version of a DBMS, location, data storage or technology to be used in the project.
- Columns should have exact data types, lengths assigned and default values.
- Primary and Foreign keys, views, indexes, access profiles, and authorizations, etc. are defined.

Advantages and Disadvantages of Data Model:

Advantages of Data model:

- The main goal of a designing data model is to make certain that data objects offered by the functional team are represented accurately.
- The data model should be detailed enough to be used for building the physical database.
- The information in the data model can be used for defining the relationship between tables, primary and foreign keys, and stored procedures.
- Data Model helps business to communicate the within and across organizations.
- Data model helps to documents data mappings in ETL process
- Help to recognize correct sources of data to populate the model

Disadvantages of Data model:

- To developer Data model one should know physical data stored characteristics.
- This is a navigational system produces complex application development, management.
- Even smaller change made in structure require modification in the entire application.
- There is no set data manipulation language in DBMS.

Database Query Language (Structured Query Language)

Structured query language (SQL) is:
- The ANSI standard language for operating relational databases
- Efficient, easy to learn, and use
- Functionally complete (With SQL, you can define, retrieve, and manipulate data in the tables

SQL COMMANDS

SELECT	Query (Data Retrieval) Language

INSERT
UPDATE
DELETE
	Data manipulation language (DML)

CREATE ALTER DROP RENAME TRUNCATE COMMENT	Data Definition Language (DDL)
GRANT REVOKE	Data Control Language (DCL)
COMMIT ROLLBACK SAVEPOINT	Transaction Control Language

Writing SQL Statements

- SQL statements are not case-sensitive.
- SQL statements can be entered on one or more lines.
- Keywords cannot be abbreviated or split across lines.
- Clauses are usually placed on separate lines.
- Indents are used to enhance readability.
- In SQL Developer, SQL statements can optionally be terminated by a semicolon (;). Semicolons are required when you execute multiple SQL statements.
- In SQL*Plus, you are required to end each SQL statement
with a semicolon (;).

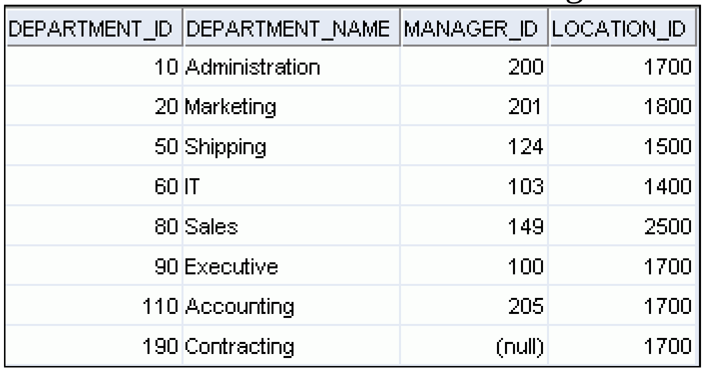
SELECT Command

The **SELECT command** is used to query the database, that is retrieve record(s) from the database tables or other database objects.

SELECT *|{[DISTINCT] *column*|*expression* [*alias*],...}

FROM *table;*

• SELECT identifies the columns to be displayed.
• FROM identifies the table containing those columns.



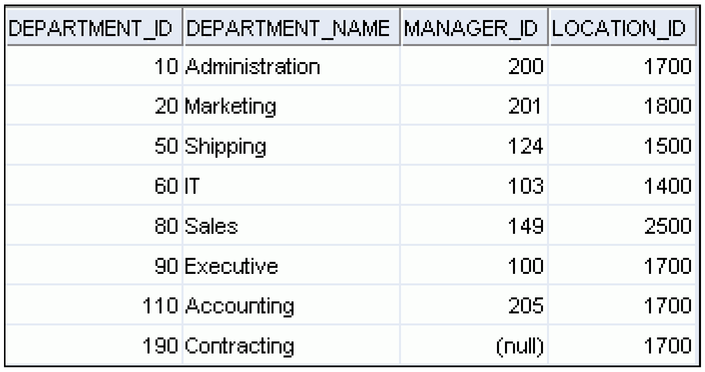
DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting	(null)	1700

Table name: departments

Selecting all columns

SELECT * FROM departments;

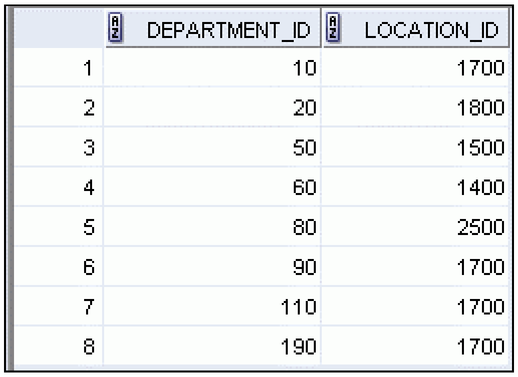
This will retrieve all rows and columns from the departments table. This means that the resulting table will have the same degree and cardinality as the original table.



DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting	(null)	1700

Selecting specific columns

SELECT department_id, location_id FROM departments;



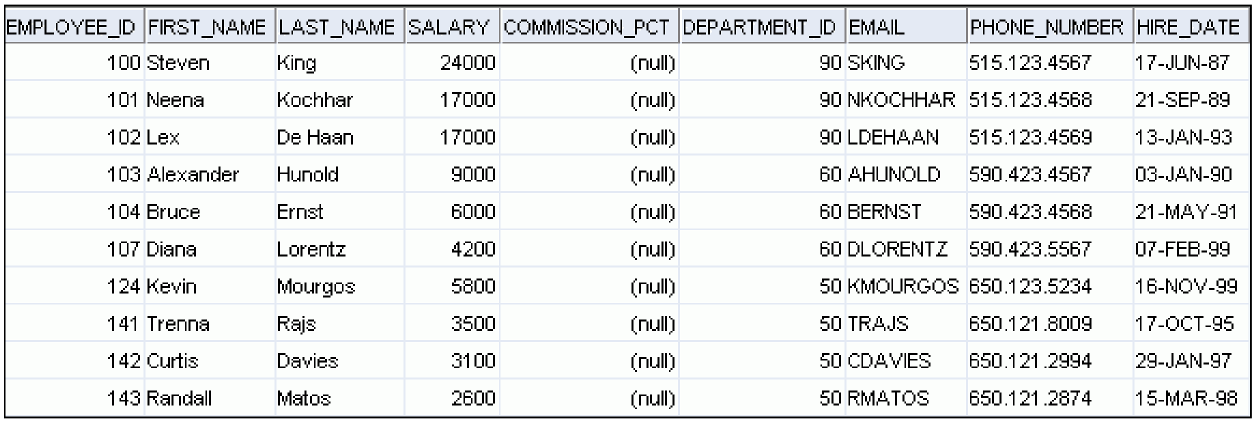
	DEPARTMENT_ID	LOCATION_ID
1	10	1700
2	20	1800
3	50	1500
4	60	1400
5	80	2500
6	90	1700
7	110	1700
8	190	1700

Restricting data

In some situations, we may not need to retrieve all rows in the table. It is possible to retrieve only rows that meet specified criteria. This is achieved using

- The WHERE clause
- The comparison conditions using =, <=, BETWEEN, IN, LIKE, and NULL conditions.
- Logical conditions using AND, OR, and NOT operators

The table below called employees will be used to demonstrate the above.



EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	COMMISSION_PCT	DEPARTMENT_ID	EMAIL	PHONE_NUMBER	HIRE_DATE
100	Steven	King	24000	(null)	90	SKING	515.123.4567	17-JUN-87
101	Neena	Kochhar	17000	(null)	90	NKOCHHAR	515.123.4568	21-SEP-89
102	Lex	De Haan	17000	(null)	90	LDEHAAN	515.123.4569	13-JAN-93
103	Alexander	Hunold	9000	(null)	60	AHUNOLD	590.423.4567	03-JAN-90
104	Bruce	Ernst	6000	(null)	60	BERNST	590.423.4568	21-MAY-91
107	Diana	Lorentz	4200	(null)	60	DLORENTZ	590.423.5567	07-FEB-99
124	Kevin	Mourgos	5800	(null)	50	KMOURGOS	650.123.5234	16-NOV-99
141	Trenna	Rajs	3500	(null)	50	TRAJS	650.121.8009	17-OCT-95
142	Curtis	Davies	3100	(null)	50	CDAVIES	650.121.2994	29-JAN-97
143	Randall	Matos	2600	(null)	50	RMATOS	650.121.2874	15-MAR-98

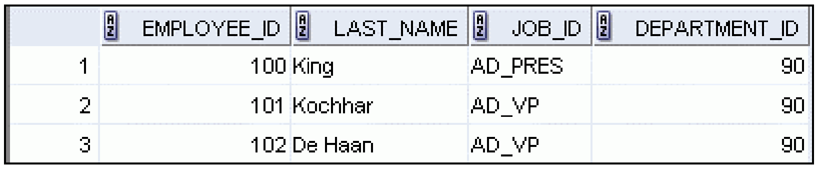
Syntax

SELECT *|{[DISTINCT] *column|expression* [*alias*],...} FROM *table* [WHERE *condition(s)*];

Example

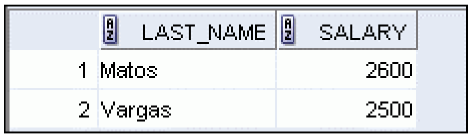
SELECT employee_id, last_name, job_id, department_id FROM employees WHERE department_id = 90 ;

When executed the above query will result in the tble below



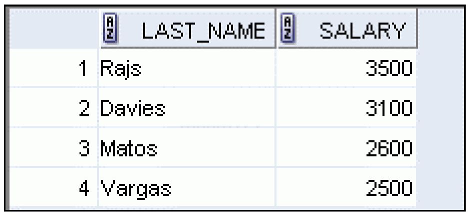
	EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	100	King	AD_PRES	90
2	101	Kochhar	AD_VP	90
3	102	De Haan	AD_VP	90

SELECT last_name, salary FROM employees
WHERE salary <= 3000 ;



	LAST_NAME	SALARY
1	Matos	2600
2	Vargas	2500

SELECT last_name, salary FROM employees WHERE salary BETWEEN 2500 AND 3500 ;



	LAST_NAME	SALARY
1	Rajs	3500
2	Davies	3100
3	Matos	2600
4	Vargas	2500

The UPDATE statement is used to modify the existing records in a table.

UPDATE Syntax

UPDATE *table_name*
SET *column1* = *value1*, *column2* = *value2*, *...*
WHERE *condition*;

Note: Be careful when updating records in a table! Notice the WHERE clause in the UPDATE statement. The WHERE clause specifies which record(s) that should be updated. If you omit the WHERE clause, all records in the table will be updated!

```
UPDATE employees SET SALARY= 10000 WHERE EMPLOYEE_ID = 104;
```

The SQL DELETE Statement

The DELETE statement is used to delete existing records in a table.

DELETE Syntax

```
DELETE FROM table_name
WHERE condition;
```

Note: Be careful when deleting records in a table! Notice the WHERE clause in the DELETE statement. The WHERE clause specifies which record(s) should be deleted. If you omit the WHERE clause, all records in the table will be deleted!

```
DELETE FROM employees WHERE EMPLOYEE_ID = 104
```

CONSTRAINTS

Constraint is the way DBMS prevents the entry of illegal data. It is the rules about how tables are related and how data are stored in the table.

Types of Table Constraints

Constraint Type	**Description**
Not Null	A value must be supplied for this column, but values do not have to be unique.
Unique Key	Every value in this column must be unique, but null values are allowed.
Primary Key	Every value in the column must be unique and cannot be null.

Foreign Key	Every value in the column must match a value in another column in this table or some other table; otherwise, the value is null.
Check	The value entered in the table must match one of the specified values for this column.

Database Auditing

Auditing is the monitoring and recording of selected user database actions. It can be based on individual actions, such as the type of SQL statement executed, or on combinations of factors that can include user name, application, time, and so on. Security policies can trigger auditing when specified elements in an Oracle database are accessed or altered, including the contents within a specified object.

Auditing is typically used to:

- Enable future accountability for current actions taken in a particular schema, table, or row, or affecting specific content
- Deter users (or others) from inappropriate actions based on that accountability
- Investigate suspicious activity

  For example, if some user is deleting data from tables, then the security administrator might decide to audit all connections to the database and all successful and unsuccessful deletions of rows from all tables in the database.

- Notify an auditor that an unauthorized user is manipulating or deleting data and that the user has more privileges than expected which can lead to reassessing user authorizations
- Monitor and gather data about specific database activities

  For example, the database administrator can gather statistics about which tables are being updated, how many logical I/Os are performed, or how many concurrent users connect at peak times.

- Detect problems with an authorization or access control implementation

  For example, you can create audit policies that you expect will never generate an audit record because the data is protected in other ways. However, if these policies do generate audit records, then you will know the other security controls are not properly implemented.

Auditing Types and descriptions

	Types	Description
1	Statement	Enables you to audit SQL statements by type of statement, not by the specific schema objects on which they operate. Typically broad, statement auditing audits the use of several types of related actions for each option. For example, `AUDIT` `TABLE` tracks several DDL statements regardless of the table on which they are issued. You can also set statement auditing to audit selected users or every user in the database.
2	Privilege	Enables you to audit the use of powerful system privileges that enable corresponding actions, such as `AUDIT` `CREATE` `TABLE`. Privilege auditing is more focused than statement auditing, which audits only a particular type of action. You can set privilege auditing to audit a selected user or every user in the database.
3	Fine-Grained	Enables you to audit at the most granular level, data access and actions based on content, using any Boolean measure, such as `value > 1,000,000`. Enables auditing based on access to or changes in a column.
4	Schema Object	Enables you to audit specific statements on a particular schema object, such as `AUDIT` `SELECT` `ON` `employees`. Schema object auditing is very focused, auditing only a single specified type of statement (such as SELECT) on a specified schema object. Schema object auditing always applies to all users of the database.

Database Security

Data is a valuable resource that must be strictly controlled and managed as with any corporate resource.

Security refers to the protection of database against unauthorised access, either intentional or accidental.

Database security:

The mechanisms that protect the database against intentional or accidental threats.

It encompasses hardware, software, people and data.

Database security refers to the collective measures used to protect and secure a database or database management software from illegitimate use and malicious threats and attacks.

It is a broad term that includes a multitude of processes, tools and methodologies that ensure security within a database environment.

Database Security can also be defined as protecting information against unauthorized disclosure, alteration or destruction. We can also define database security as the mechanisms that protect the databases against intentional or accidental threats.

Database security covers and enforces security on all aspects and components of databases. This includes:

- Data stored in database
- Database server
- Database management system (DBMS)
- Other database workflow applications

Database security aims to minimize losses caused by anticipated events in a cost effective manner without unduly constraining the users.

The complete solution to data security must meet the three main requirements of *Confidentiality, Integrity and Availability*.

Confidentiality - protecting information from being disclosed to unauthorised parties.
integrity - protecting information from being changed by unauthorised parties.

Availability - to the availability of information to authorised parties only when requested.
Database security is generally planned, implemented and maintained by a database administrator and or other information security professional.

Some of the ways database security is analyzed and implemented include:

- Restricting unauthorized access and use by implementing strong and multifactor access and data management controls
- Load/stress testing and capacity testing of a database to ensure it does not crash in a distributed denial of service (DDoS) attack or user overload

- Physical security of the database server and backup equipment from theft and natural disasters
- Reviewing existing system for any known or unknown vulnerabilities and defining and implementing a road map/plan to mitigate them

Authentication is a process in which the credentials provided are compared to those on file in a database of authorized users' information on a local operating system or within an authentication server. If the credentials match, the process is completed and the user is granted authorization for access.
It is the process of determining whether someone or something is, in fact, who or what it is declared to be.

An **authentication factor** is a category of credential used for identity verification. The three most common categories are often described as something you know (the knowledge factor), something you have (the possession factor) and something you are (the inherence factor).

Authentication factors:

- **Knowledge factors** -- a category of authentication credentials consisting of information that the user possesses, such as a personal identification number (PIN), a user name, a password or the answer to a secret question.
- **Possession factors** -- a category of credentials based on items that the user has with them, typically a hardware device such as a security token or a mobile phone used in conjunction with a software token.
- **Inherence factors** -- a category of user authentication credentials consisting of elements that are integral to the individual in question, in the form of biometric data.

Authorization is the process of giving someone permission to do or have something.

Access control
Access controls for a database system is based on the granting and revoking privileges.
A privilege allows a user to create or access some database object or to run certain DBMS utilities.

Discretionary security mechanisms

- Privilege grants allow specific users to perform specific operations on specific data
- Initial grants start with DBA
- Grants may be passed on between users

Mandatory security mechanisms

- Enforce multi-level security
- Data and users are classified into security classes
- Typically, user can only see data which has a lower (or same) classification as themselves
- Role-based security is similar

Backup and Recovery

- Backup is the process of periodically taking a copy of the database and log file (and possibly programs) on to offline storage media.
- A DBMS should provide facilities to assist with the recovery of database following failure.
- ***Journaling*** is the process of keeping and maintaining a log file (or journal) of all changes made to the database to enable recovery to be undertaken effectively in the event of failure.