Structured Query Language

SQL History

- 1973 - systems developed to test the relational model
  - System R developed by IBM
  - Ingres (Stonebraker)

SEQUEL

- Structured English QUEry Language
  - implemented by System R
  - easy to learn & use
  - based on familiar English keywords & avoided difficult relational algebra concepts such as the division operator
  - also included update operators and a few other concepts such as view creation & definition of constraints

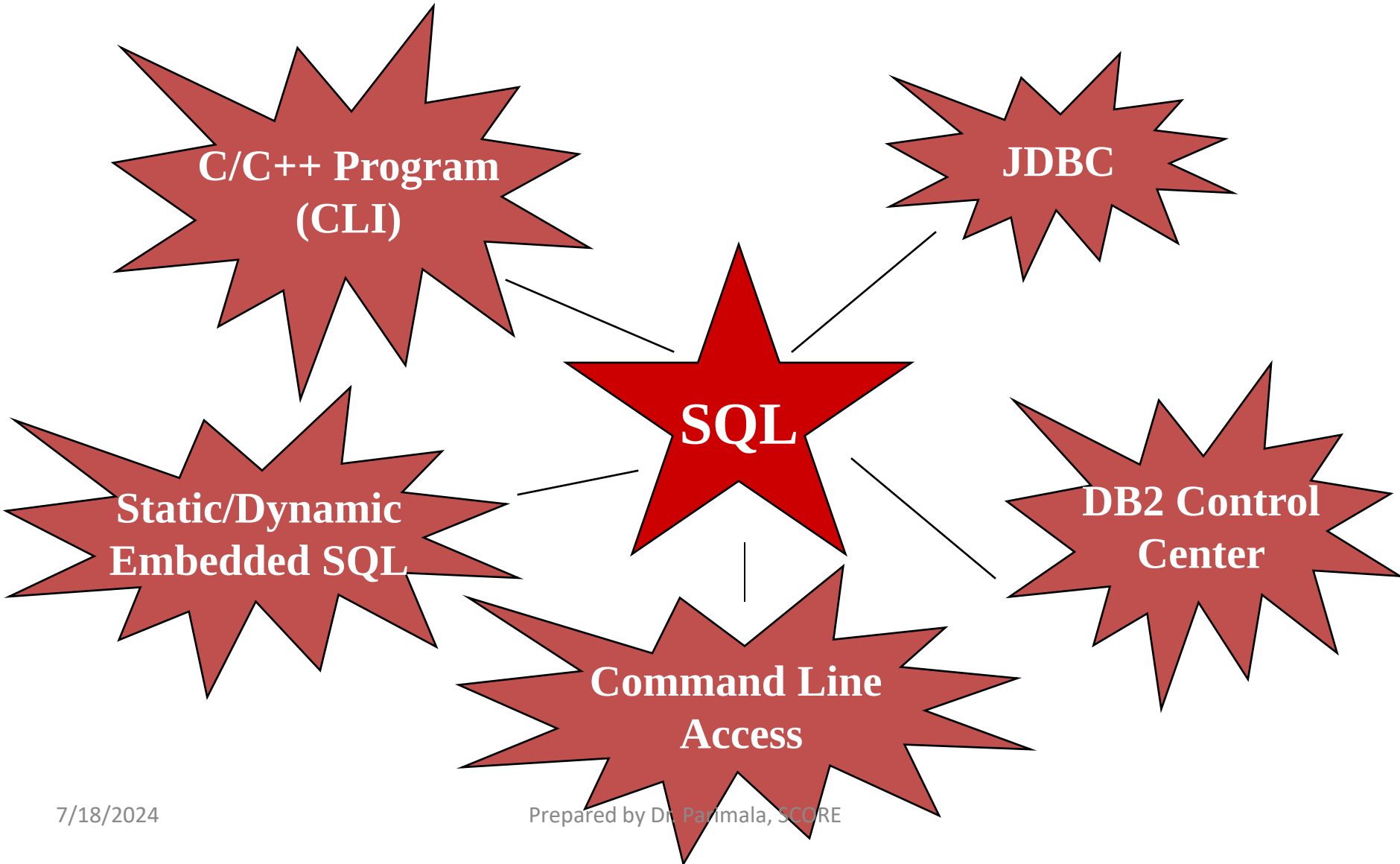
Late ‘70’s

- SEQUEL was found to be a trademark for something else …
  - so, the name was changed to SQL: Structured Query Language
- First product based on SQL was called Oracle (1979) by a company called Relational Software, Inc.

1980's

- IBM gets in the game in 1981 with SQL/Data System
- SQL was implemented by all major relational database suppliers & is now the world's most widely used database language.
- ANSI project to develop SQL standards
  - ANSI SQL, SQL-92 (SQL2) and SQL3 but...
    - every Relational DMBS implements a different level of SQL -- just to confuse you!

SQL is the backbone



One preliminary note..

- SQL is case **in**sensitive!!

  CREATE TABLE = create table = CreAte TAblE

- Quoted text, however, is case sensitive

  “Payroll” != “payroll”

1 Introduction to SQL

What is SQL?

- **When a user wants to get some information from a database file, he can issue a *query*.**
- **A query is a user–request to retrieve data or information with a certain condition.**
- **SQL is a query language that allows user to specify the conditions. (instead of algorithms)**

1 Introduction to SQL

Concept of SQL

- **The program will go through all the records in the database file and select those records that satisfy the condition.(searching)**

- **The result of the query will then be stored in form of a table.**

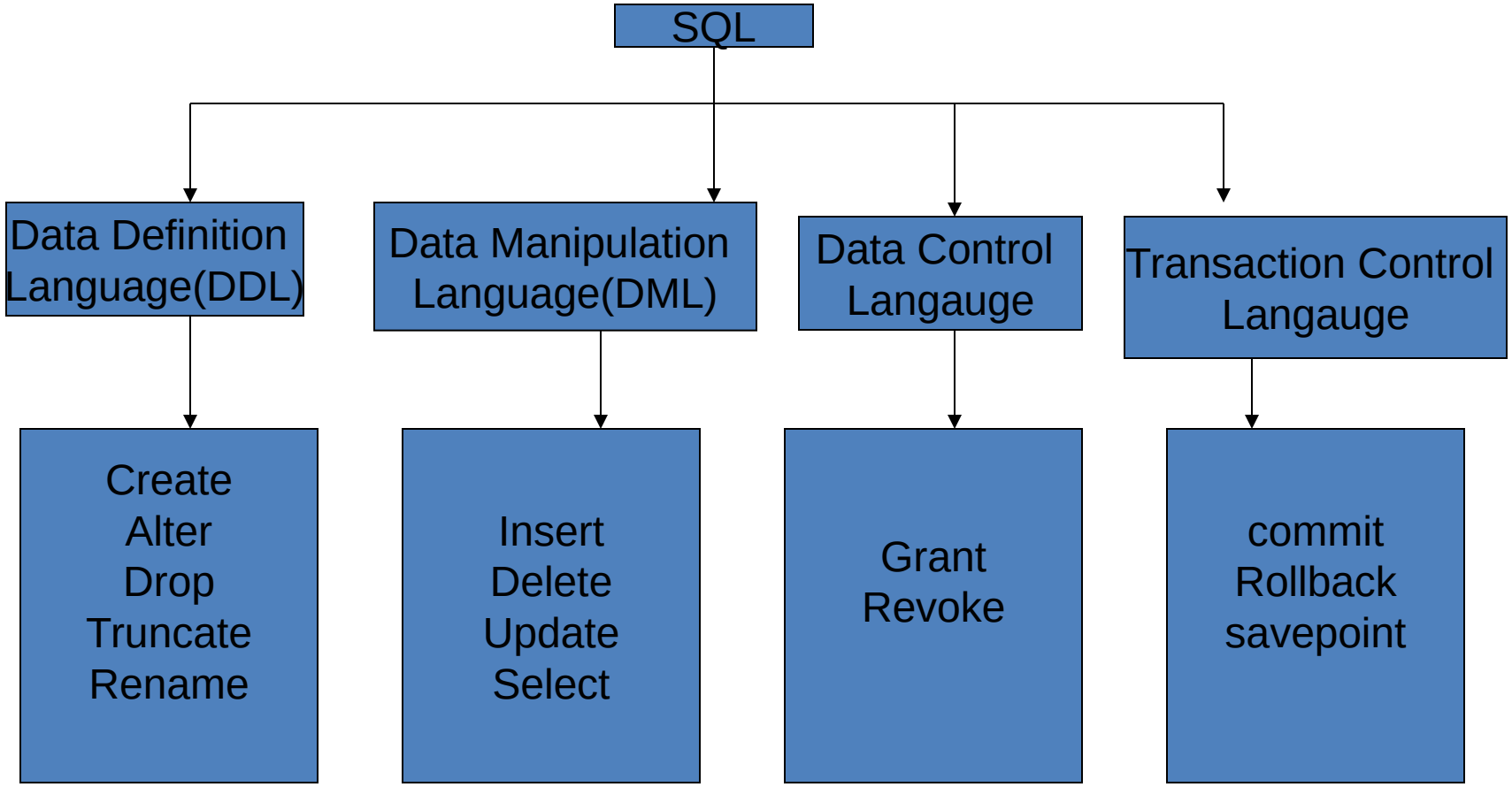
2 Basic structure of an SQL query

General Structure	SELECT, ALL / DISTINCT, *, AS, FROM, WHERE
Comparison	IN, BETWEEN, LIKE "% _"
Grouping	GROUP BY, HAVING, COUNT( ), SUM( ), AVG( ), MAX( ), MIN( )
Display Order	ORDER BY, ASC / DESC
Logical Operators	AND, OR, NOT
Output	INTO TABLE / CURSOR TO FILE [ADDITIVE], TO PRINTER, TO SCREEN
Union	UNION

DATA TYPES

- Character
  - char (length)
  - ex: char(10)
- Character
  - varchar2(length)
  - ex: varchar2(5)
- Number
  - number(3) eg., values should be 123, 467
  - number (4,1) eg., values should be 123.4, 124.6, 22.5
- Date
  - Date
  - Eg., 22-oct-82 or 22-oct-1982

SQL COMMANDS



1. CREATE-Syntax 1

a. simple creation

CREATE TABLE < tablename> (
<column name1 > < datatype>,
<column name 2> < datatype>,
<column name 3> < datatype>);

Create table book(
Bookno number(5),
Bookname varchar2(20),
Authorname varchar2(25),
Dop date,
Price number(6,2));

CREATE-Syntax 2

Table-level Primary key constraint

```
CREATE TABLE  < tablename>   (
     <column name 1>          < datatype>,
     <column name 2>          < datatype> unique ,
     <column name 3>          < datatype> ,
     primary key ( <column name2>));

Create table book(
Bookno number(5),
Bookname varchar2(20),
Authorname varchar2(25),
Dop date,
Price number(6,2),
Primary key (Bookno));
```

Row-level Primary key constraint

CREATE TABLE < tablename> (
    <column name1 > < datatype> primary key,
    <column name 2> < datatype>,
    <column name 3> < datatype>);

Create table book(
Bookno number(5) primary key,
Bookname varchar2(20),
Authorname varchar2(25),
Dop date,
Price number(6,2));

CREATE-Syntax 3 with constraint name and Foreign Key

with constraint name and FK

CREATE TABLE < tablename1> (
<column name 1> < datatype>,
<column name 2> < datatype>,
constraint < constraint name1 > primary key (<column name1>),
constraint <constraint name2> foreign key (<column name2>)
references <tablename2> (<column name1>));

Create table emp(
Eno number(3),ename char(30), dn number(2),
Constraint emp_pk Primary key(eno),
Constraint emp_fk Foreign key(dn) references dept(dno));

Create table dept(dno number(2) primary key, dname char(20));

Depend(dname, empno)
Primary key(dname, empno)
Foreign key(empno) references emp(eno)

Create table emp(Eno number(3),ename char(30), dn number(2), pno number(3),
Constraint emp_pk Primary key(eno),
Constraint emp_fk Foreign key(dn) references dept(dno)
Constraint emp_fk2 Foreign key(pno) references project(projno)
Check constrint);
101 sdf 10
102 uou 20
103 nbn 30

Create table project(projno number(3), pname varchar(20), primary key(projno));
Create table dept(dno number(2) primary key, dname char(20));
10 as
20 er

Rules for Foreign key

Common column

Same domain

Parent- pk(unique) / child – fk(duplicates)

Fk column names can be different or same

CREATE-Syntax 4 with check constraint

with check constraint

```
CREATE TABLE  < tablename>   (
<column name1 >        < datatype> ,
<column name 2>        < datatype>,
check  ( < column name 1  > in  ( values) )
check  ( < column name 2  >  between <val1> and  <val2> ) );
```

- Check (bookname in (‘prog in c’, ‘DBMS’,’Data structures’))
- Check (price between 0 and 5000)

2. ALTER

a. Add –to add new columns

ALTER TABLE <tablename> add (<column name > < datatype>)
Eg., alter table emp add(SAL NUMBER(6));

b. Modify the datatype or increase / decrease the column width

ALTER TABLE <tablename> modify (<column name > <newdatatype>)
EG., alter table emp modify(sal number(4));

c. drop –delete column or remove constraint

ALTER TABLE <tablename> drop column < column name>;
Eg., alter table emp drop column sal;
ALTER TABLE <tablename> drop constraint < constraint name > ;
Eg., alter table emp drop constraint emp_fk;

***** Constraints addition and column changing (datatype or decreasing the width) can be done only if column values are null.**

3. TRUNCATE

- Removes the rows, not the definition
- TRUNCATE TABLE <tablename>;

4. DROP

- Removes the rows and table definition
- DROP TABLE <tablename>;

5. RENAME

- Changes table name
- RENAME < old tablename> to < new tablename>;

Practise-1

Emp(empno,ename,age,sal,dno)

Dept(depno,dname)

Use all the create syntax

Use all the alter command

Use truncate,drop and rename