Q a O J

Chapter 5

A

1	Introduction of Structured Query Language
2	DDL: create, drop, alter
3	DML: select, insert, update, delete
4	DCL: commit, rollback, grant, revoke

A

	G a Q a O J
2	DDL: create, drop, alter
3	DML: select, insert, update, delete
4	DCL: commit, rollback, grant, revoke

G a Q a O J

- **Q a O J** (SQL) is a standard computer language for relational database management and data manipulation.
- Basic SQL:
  - Data Definition Language (DDL)
    - Create, Alter, Drop
  - Data Manipulation Language (DML)
    - Select, Insert, Update, Delete
  - Data Control Language (DCL)
    - Commit, Rollback, Grant, Revoke

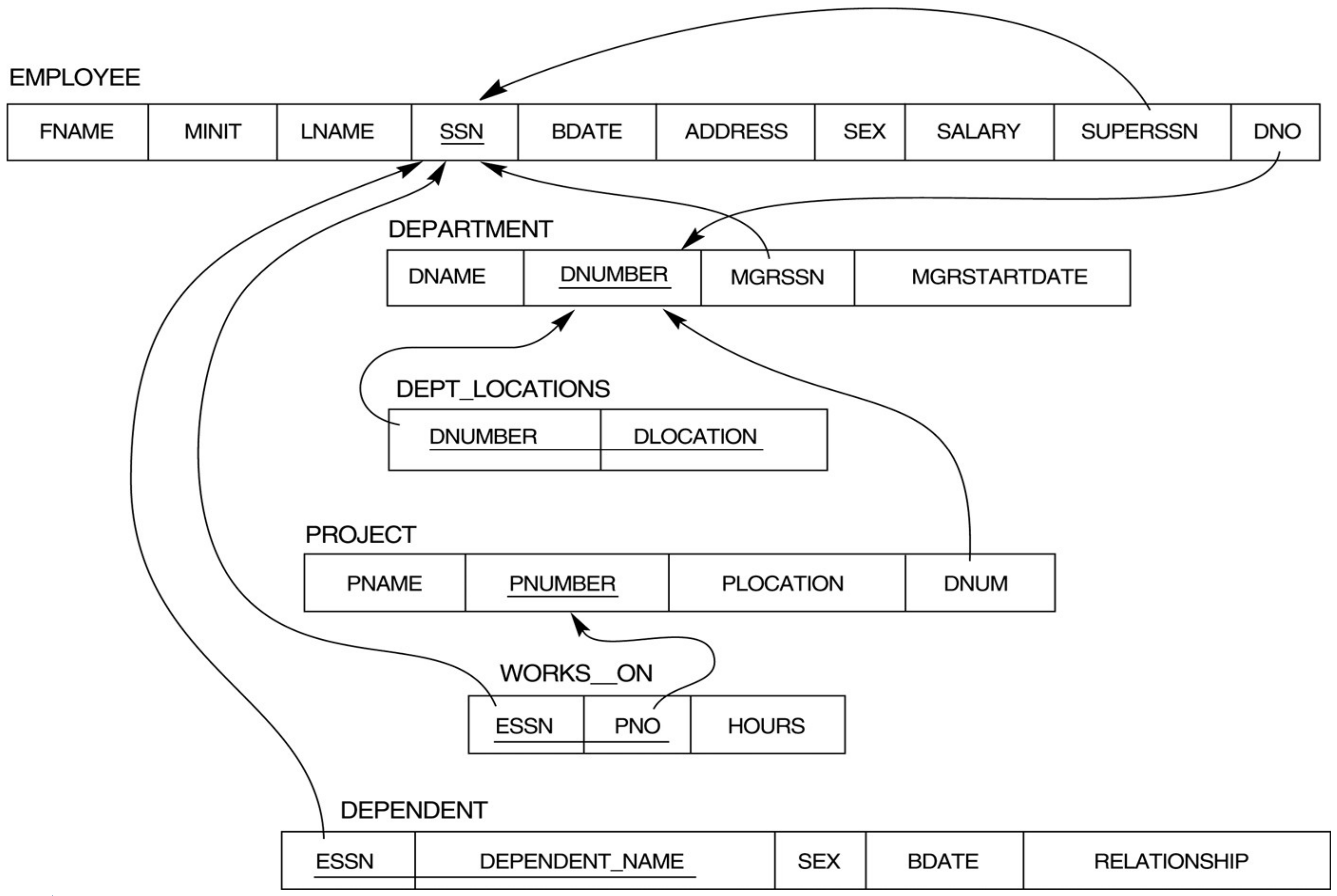
A

1	Introduction of Structured Query Language
	BBJ a
3	DML: select, insert, update, delete
4	DCL: commit, rollback, grant, revoke

B B J BBJ

- Permits specification of data types, structures and any data constraints
- All specifications are stored in the database
- Includes:
  - **APC RC**: make a new database object (database, table, index, user, stored query, …)
  - **JRCP**: modify an existing database object
  - **BPMN**: destroy an existing database object

R AMK N L WB



EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

Qa A A a QOJ

- **Qa** : a group of tables and other constructs that belong to the same database application

CREATE SCHEMA *Schema_Name*
AUTHORIZATION *Authorization_Identifier*;

APC RC QAF CK Company **S RF MPGX RGML** JSmith;

- **A** : a named collection of schemas

APC RC R J C

CREATE TABLE [*SchemaName.*]*TableName*

({*colName dataType* [NOT NULL] [UNIQUE] [PRIMARY KEY]

[DEFAULT *defaultOption*]

[CHECK *searchCondition*] [,...]}

[PRIMARY KEY (*listOfColumns*),]

{[UNIQUE (*listOfColumns*),] [...,]}

{[FOREIGN KEY (*~~listOf~~FKColumns*)

REFERENCES *ParentTableName* [(*~~listOf~~CKColumns*)]

[ON UPDATE *referentialAction*]

[ON DELETE *referentialAction*]] [,...]}

{[CHECK (*searchCondition*)] [,...] })

APC RC R J C

- (base relations)
  - Relation and its tuples are actually created and stored as a file by the DBMS.
- **T**
  - Created through the CREATE VIEW statement.
- Some foreign keys may cause errors
  - Circular references
  - refer to a table that has not yet been created

a B R

- **L a :**
  - Integer numbers: INTEGER, INT, and SMALLINT
  - Floating-point (real) numbers: FLOAT or REAL, and DOUBLE PRECISION
- **A a :**
  - Fixed length: CHAR(n), CHARACTER(n)
  - Varying length: VARCHAR(n), CHAR VARYING(n), CHARACTER VARYING(n)
- **:**
  - Fixed length: BIT(n)
  - Varying length: BIT VARYING(n)
- **:**
  - Values of TRUE or FALSE or NULL
- **B R :**
  - Date components: YEAR, MONTH, and DAY (‘YYYY-MM-DD’)
  - Time components: HOUR, MINUTE, and SECOND (‘HH:MM:SS’)

a B R

- Additional data types
  - **R RGK CQR K N**
    - Includes the DATE and TIME fields
    - Plus a minimum of six positions for decimal fractions of seconds
    - Optional WITH TIME ZONE qualifier
  - **GL RCPT J**
    - Specifies a relative value that can be used to increment or decrement an absolute value of a date, time, or timestamp

B

- Name used with the attribute specification
- Makes it easier to change the data type for a domain that is used by numerous attributes
- Improves schema readability

CREATE DOMAIN *DomainName* AS *DataType*
[CHECK *conditions*];

```
APC  RC BMK  GL SSN_TYPE  QCHAR(9);
APC  RC BMK  GL D_NUM  QINTEGER
AFCAI  (D_NUM>0 AND D_NUM<21);
```

Q a A

- Basic constraints:
  - Key and referential integrity constraints
  - Attribute constraints
  - Constraints on individual tuples within a relation

I P G A

- **NPGK PWI CW** clause: specifies one or more attributes that make up the primary key of a relation.

Dnumber INT **NPGK PWI CW**

NPGK PWI CW(Dnumber, DLocation)

- **S L GOS C** clause: Specifies alternate (secondary) keys.

Dname VARCHAR(15) **S L GOS C**;

I P G A

- **DMPCGEL I CW** clause

FOREIGN KEY (*listOfFKColumns*)
REFERENCES *ParentTableName* [(*listOfCKColumns*)]
[ON UPDATE *referentialAction*]
[ON DELETE *referentialAction*]

- Referential triggered actions: RESTRICT (default), SET NULL, CASCADE, and SET DEFAULT

DMPCGEL I CW Dno **PCDCPCL ACQ** Department(Dnumber)
ML BCJ CRC CASCADE
ML S NB RC CASCADE

A

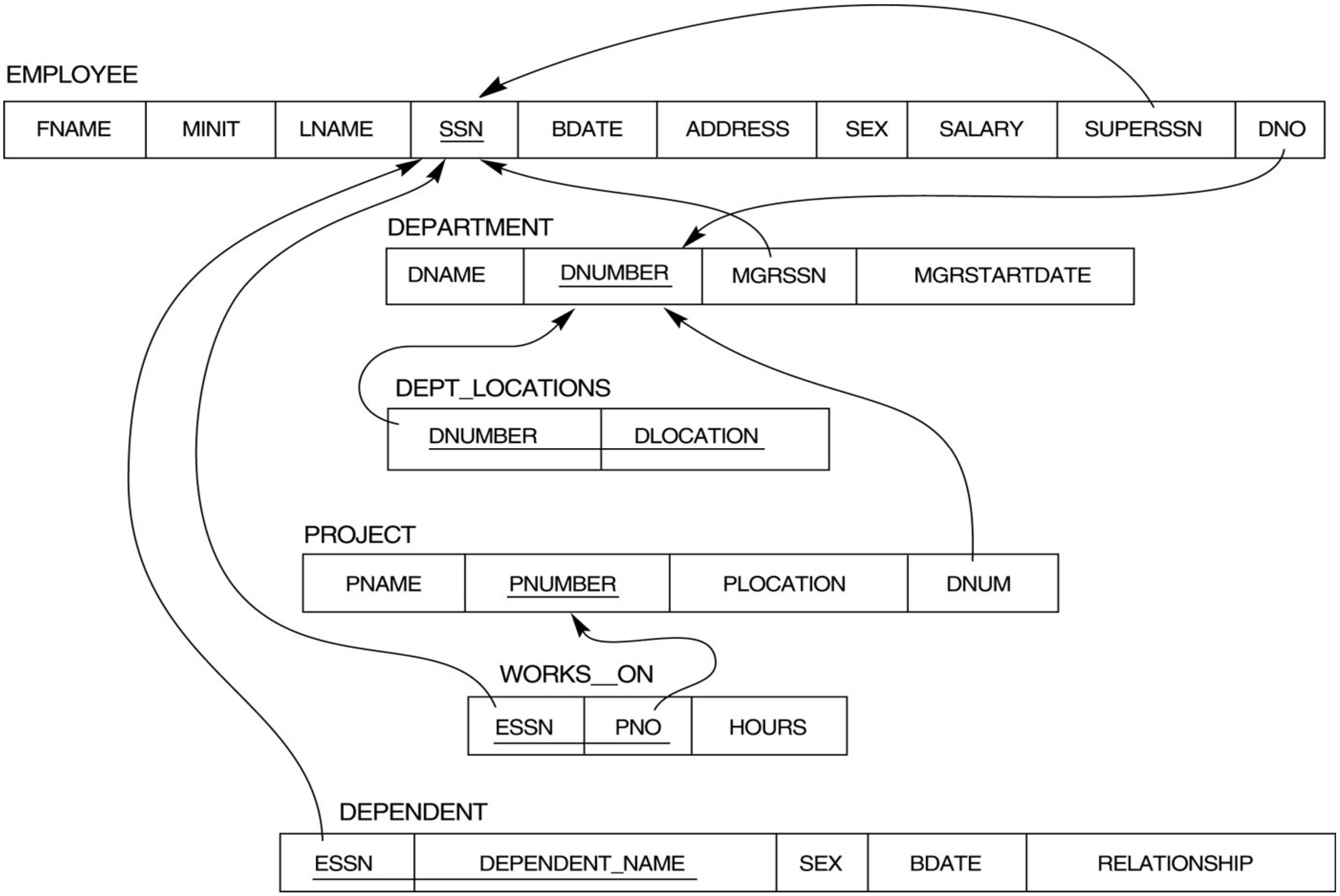
- NOT NULL
  - NULL is permitted for a particular attribute
- Default values
  - DEFAULT <value> can be specified for an attribute
  - If no default clause is specified, the default value is NULL for attributes that do not have the NOT NULL constraint

Dno INT **L MR L S J J BCD S JR**

- CHECK clause:

Dnumber INT NOT NULL **AF CAI B L B**
B ;

R AMK N L WB


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

```
CREATE TABLE EMPLOYEE
        ( Fname                     VARCHAR(15)             NOT NULL,
          Minit                     CHAR,
          Lname                     VARCHAR(15)             NOT NULL,
          Ssn                       CHAR(9)                 NOT NULL,
          Bdate                     DATE,
          Address                   VARCHAR(30),
          Sex                       CHAR,
          Salary                    DECIMAL(10,2),
          Super_ssn                 CHAR(9),
          Dno                       INT                     NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                     VARCHAR(15)             NOT NULL,
          Dnumber                   INT                     NOT NULL,
          Mgr_ssn                   CHAR(9)                 NOT NULL,
          Mgr_start_date            DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber                   INT                     NOT NULL,
          Dlocation                 VARCHAR(15)             NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
```

```
CREATE TABLE PROJECT
        ( Pname                     VARCHAR(15)                 NOT NULL,
          Pnumber                   INT                         NOT NULL,
          Plocation                 VARCHAR(15),
          Dnum                      INT                         NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                      CHAR(9)                     NOT NULL,
          Pno                       INT                         NOT NULL,
          Hours                     DECIMAL(3,1)                NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                      CHAR(9)                     NOT NULL,
          Dependent_name            VARCHAR(15)                 NOT NULL,
          Sex                       CHAR,
          Bdate                     DATE,
          Relationship              VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Q a A

- Giving names to constraints
  - This is optional.
  - Keyword **AML QRP GLR**
  - The name is unique within a particular DB schema.
  - Used to identify a particular constraint in case it must be dropped later and replaced with another one.

```
CREATE TABLE EMPLOYEE
    ( ... ,
      Dno            INT              NOT NULL       DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                    ON DELETE SET NULL          ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                    ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( ... ,
      Mgr_ssn CHAR(9)           NOT NULL        DEFAULT '888665555',
      ... ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
IGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
        ON DELETE SET DEFAULT       ON UPDATE CASCADE);        CREATE T
BLE DEPT_LOCATIONS                                             ( ... ,
                                                                PRIM
RY KEY (Dnumber, Dlocation),                                    FORE
GN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
      ON DELETE CASCADE             ON UPDATE CASCADE);
```

A

- Specifying constraints on tuples using CHECK
  - Affected on each tuple individually as being inserted or modified (tuple-based constraints)
  - Ex: Department's create-date must be earlier than the manager's start-date:

```
AF CAI  (DEPT_CREATE_DATE < MGRSTARTDATE);
```

- More general constraints: CREATE ASSERTION

BPMN A

- Used to drop named schema elements: tables, domains, constraints, and the schema itself
- Drop behavior options:
  - CASCADE and RESTRICT

BPMN QAF CK Company **A QA BC**;

Or

BPMN QAF CK Company **PCQRPGAR**;

BPMN A

- Drop a table:

BPMN R J C Department **A QA BC**;

  - RESTRICT (default): dropped on if it is not referenced in any constraints or views
  - CASCADE: all such constraints and views that reference the table are dropped automatically from the schema along with the table itself
- Similarly, we can drop constraints & domains

JRCP A

- ALTER command: change the definition of a base table or of other named schema elements
- Base tables: adding or dropping a column or constraints, changing a column definition.

```
JRCP R    J C Employee   BB Job VARCHAR(15);
JRCP R    J C Employee
     BPMN AMJ S K L Address CASCADE;
JRCP R    J C Department
       JRCP AMJ S K L Mgr_ssn SET DEFAULT ‘333445555’;
JRCP R    J C Employee
     BPMN AML QRP  GL R Empsuperfk CASCADE;
JRCP R    J C Employee
     Foreign key Dno references Department(Dnumber);
```

A

1	Introduction of Structured Query Language
2	DDL: create, drop, alter
	BKJ a
4	DCL: commit, rollback, grant, revoke

QCJ CAR A

- SELECT command: retrieve information from a database
- SELECT command in SQL is the same as the SELECT operation in relational algebra.
- SQL allows a table (relation) to have two or more tuples that are identical in all their attribute values
- SQL relation (table) is a multi-set (sometimes called a bag) of tuples; it is not a set of tuples
- SQL relations can be constrained to be sets by specifying PRIMARY KEY or UNIQUE attributes, or by using the DISTINCT option in a query

QCJ CAR A

- Basic form:

```
SELECT      <attribute list>
FROM        <table list>
WHERE       <condition>
```

- <attribute list> is a list of attribute names whose values are to be retrieved by the query
- <table list> is a list of the relation names required to process the query
- <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query

QCJ CAR A

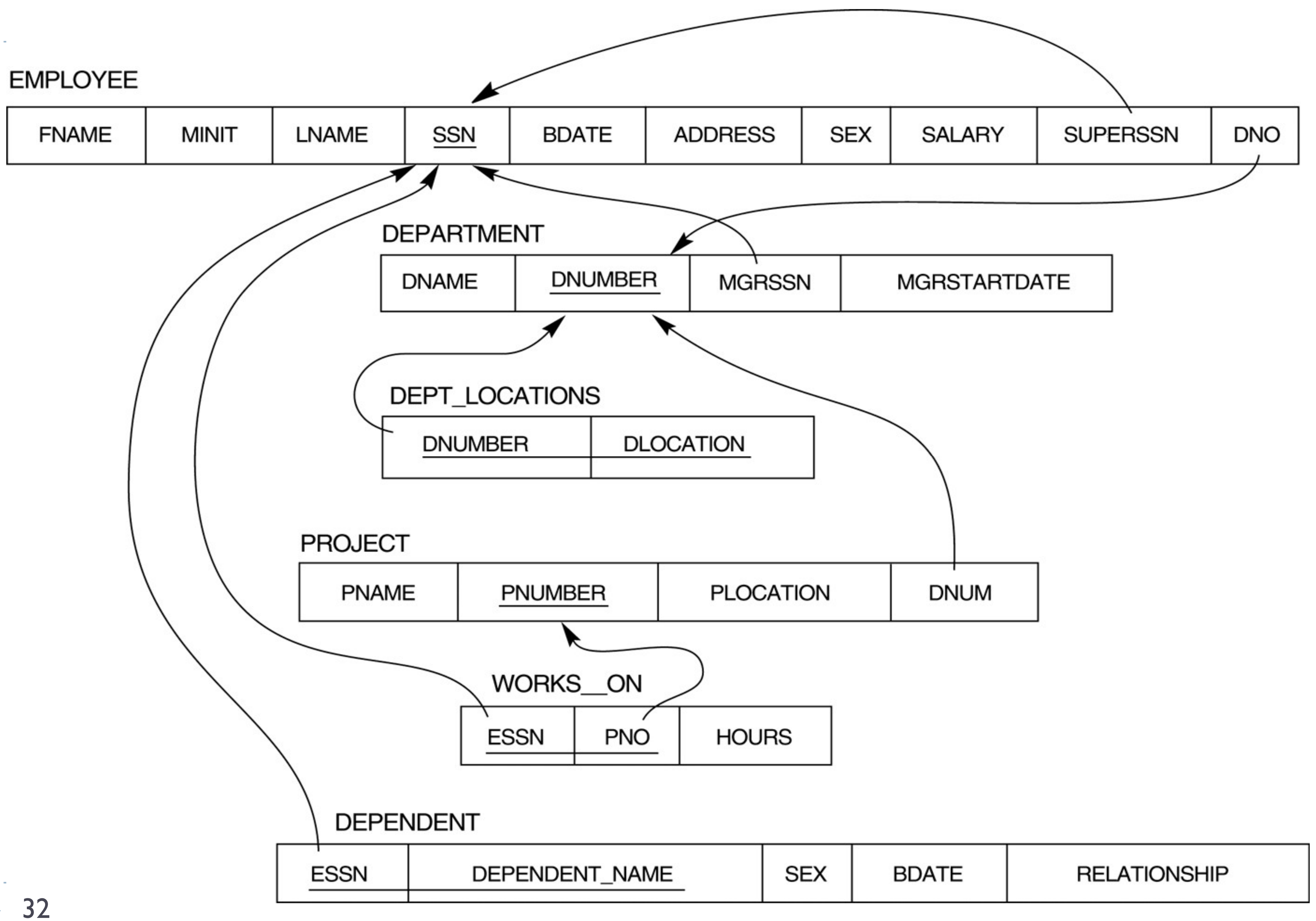
- Complete form:

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM TableName [alias] [, ...]
[WHERE condition]
[GROUP BY columnList]    [HAVING condition]
[ORDER BY columnList]
```

QCJ CAR A

SELECT	: Specifies which columns are to appear in output
FROM	: Specifies table(s) to be used
WHERE	: Filters rows
GROUP BY	: Forms groups of rows with same column value
HAVING	: Filters groups subject to some condition
ORDER BY	: Specifies the order of the output

R AMK N L WB


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

QCJ CAR A

- Basic SQL queries: using the SELECT, PROJECT, and JOIN operations of the relational algebra

Query 0: Retrieve the birthdate and address of the employee whose name is 'John B. Smith'.

```
Q0:   QCJCAR  Bdate, Address
      DPMK    Employee
      U FCPC  Fname = 'John'   LB Minit = 'B'
                LB Lname = 'Smith';
```

- Similar to a SELECT-PROJECT pair of relational algebra operations:
  - SELECT clause specifies the projection attributes
  - WHERE clause specifies the selection condition
  - However, the result of the query may contain a

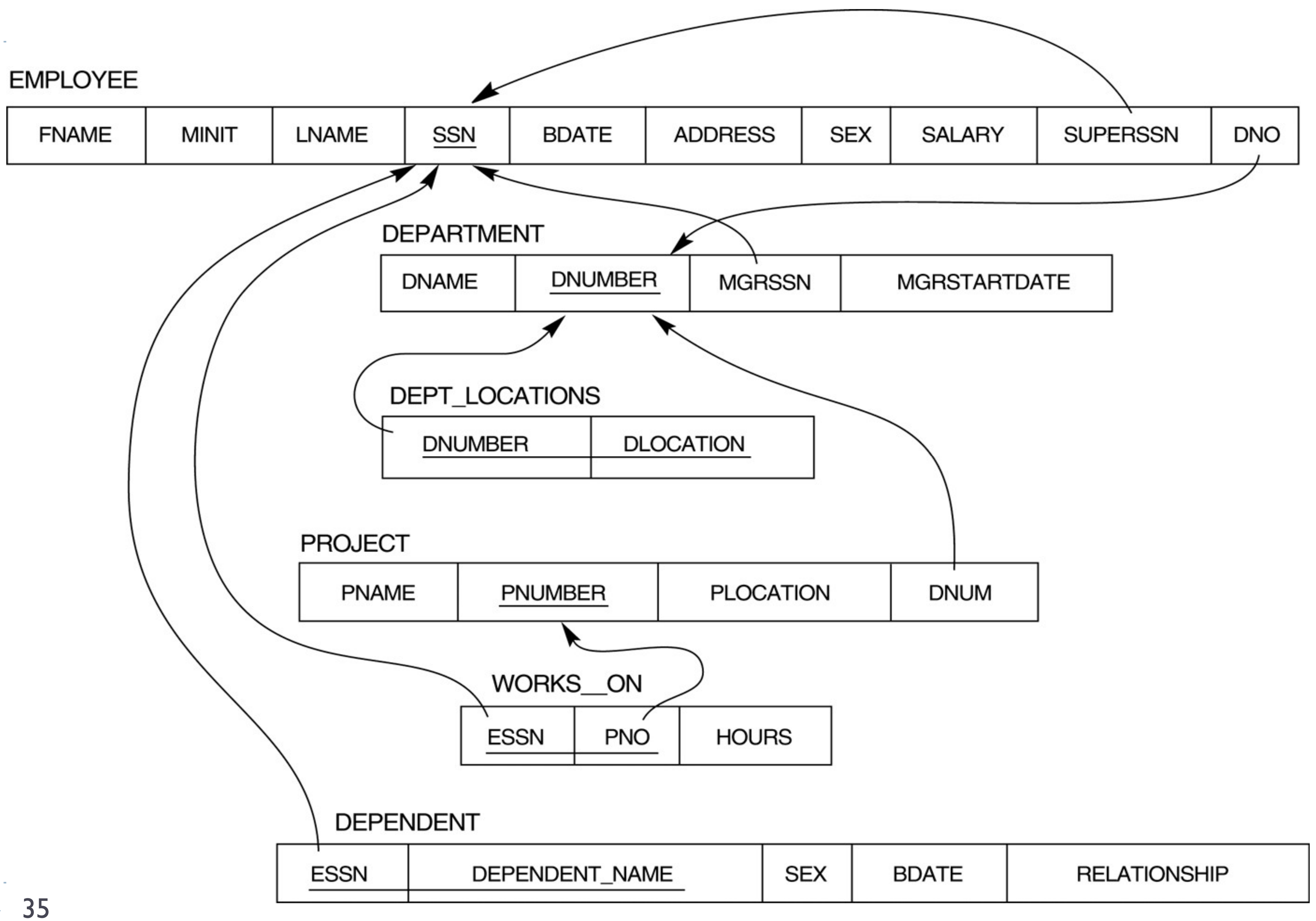
QCJ CAR A

Query 1: Retrieve the name and address of all employees who work for the 'Research' department.

```
Q1:   QCJ CAR   Fname, Lname, Address
      DPMK      Employee, Department
      U F CPC   Dname='Research'   L B  Dnumber= Dno;
```

- Similar to a SELECT-PROJECT-JOIN sequence of relational algebra operations
  - (DNAME='Research'): selection condition (SELECT operation in relational algebra)
  - (DNUMBER=DNO): join condition (JOIN operation in relational algebra)

R AMK N L WB


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

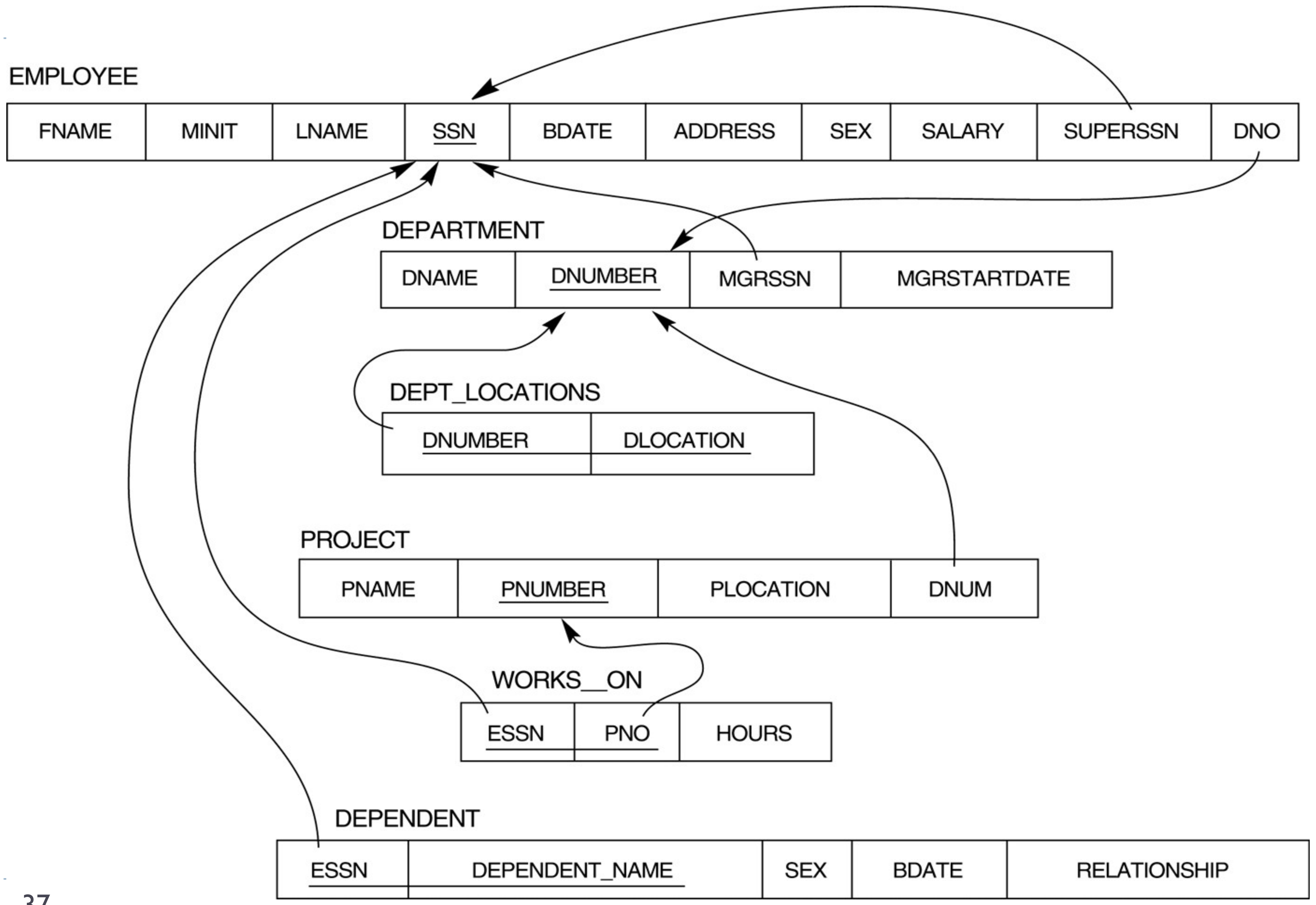
QCJ CAR A

Query 2: For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birthdate

```
Q2:   QCJ CAR  Pnumber, Dnum, Lname, Bdate, Address
      DPMK     Project, Department, Employee
      U F CPC  Dnum = Dnumber    L B MgrSSN = SSN
                   L B Plocation='Stafford‘;
```

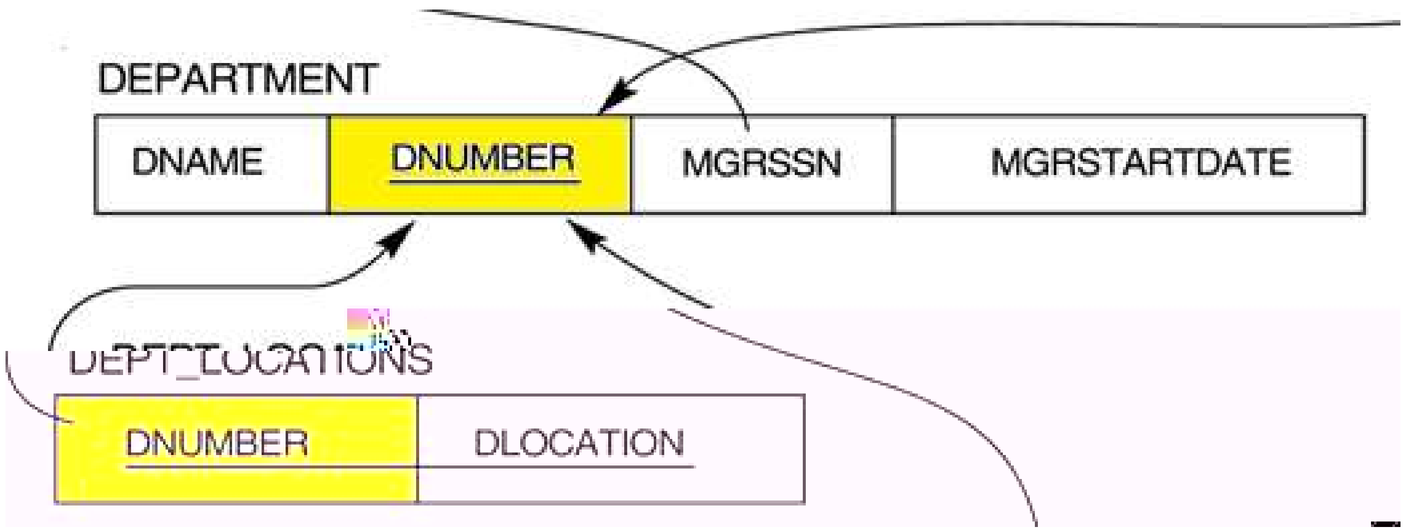
- Two join conditions:
  - Dnum = Dnumber: relates a project to its controlling department
  - MgrSSN = SSN: relates the controlling department to the employee who manages that department

R AMK N L WB


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

L

- In SQL, we can use the same name for attributes as long as the attributes are in *different relations*. Query referring to attributes with the same name must *qualify* the attribute name with the relation name by *prefixing* the relation name to the attribute name
- Examples:
  - DEPARTMENT.DNUMBER and DEPT_LOCATIONS.DNUMBER



- Some queries need to refer to the same relation twice: aliases are given to the relation name

Query 3: For each employee, retrieve the employee's name, and the name of his or her immediate supervisor.

Q3a: **QCJ CAR** E.Fname, E.Lname, S.Fname, S.Lname
DPMK Employee E S
U F CPC E.SuperSSN = S.SSN;

- E and S are called aliases or tuple variables for the Employee relation
  - E represents employees in role of supervisees
  - S represents employees in role of supervisors

- Aliases can also be used in any SQL query for convenience. Can also use the AS keyword to specify aliases

Q3b: **QCJ CAR** E.Fname, E.Lname, S.Fname, S.Lname
DPMK Employee **Q** E, Employee **Q** S
U F CPC E.SuperSSN = S.SSN;

- Renaming using aliases:

Employee **Q** E(FN, M, LN, SSN, BD, Addr, Sex, Sal, SSSN, DNO)

S a U FCPC a

- A missing WHERE-clause indicates no condition: all tuples of the relations in the FROM-clause are selected
- This is equivalent to the condition WHERE TRUE

Query 4: Retrieve the SSN values for all employees

```
Q4:   QCJCAR  SSN
      DPMK    Employee;
```

S a U F CPC a

- If more than one relation is specified in the FROM-clause and there is no join condition, then the CARTESIAN PRODUCT of tuples is selected

Query 5: retrieve all combinations of Employee.SSN and Department.Dname

Q5: **QCJCAR** SSN, Dname
DPMK Employee, Department;

- It is extremely important not to overlook specifying any selection and join conditions in the WHERE-clause; otherwise, incorrect and very large relations may result

S QRCPQQI

- An asterisk (*) stands for *all the attributes*

Query 6: retrieves all the attribute values of any Employee who works in Department number 5

```
Q6:   QCJCAR   *
      DPMK     Employee
      U F CPC  DNO = 5;
```

Query 7: retrieves all the attributes of an Employee and the attributes of the Department in which he or she works for every employee of the 'Research' department

```
Q7:   QCJCAR   *
      DPMK     Employee, Department
      U F CPC  Dname = 'Research'  LB DNO = Dnumber;
```

S BCQRGL AR

- SQL does not treat a relation as a set: duplicate tuples can appear in a query result.
- To eliminate duplicate tuples, use the keyword DISTINCT

Query 8: Retrieve the salary of every employee (Q8A) and all distinct salary values (Q8B)

Q8a: **QCJ CAR** Salary
DPMK Employee;

Q8b: **QCJ CAR** **BCQRGL AR** Salary
DPMK Employee;

- The result of Q8A may have duplicate SALARY values, but Q8B's

C

1. Retrieve the names of all employees in the departments which are located in Houston
2. List the names of all employees who have a dependent with the same first name as themselves
3. Make a list of all project numbers for projects that involve an employee whose last name is 'Smith' as a as a manager of the department that controls the project.

Q M

- Set union **SLGML** , set difference (**CVACNR** and set intersection (**GLRCPQCAR** operations
- The resulting relations of these set operations are sets of tuples: *duplicate tuples are eliminated from the result*
- The set operations apply only to *union compatible relations*
- UNION ALL, EXCEPT ALL, INTERSECT ALL

Q M

Query 9: Make a list of all project numbers for projects that involve an employee whose last name is 'Smith' as a worker or as a manager of the department that controls the project.

```
Q10:  (QCJ CAR BGQRGL AR Pnumber
       DPMK     Project, Department, Employee
       U F CPC   Dnum = Dnumber   L B MgrSSN = SSN
                  L B Lname = 'Smith')
       S L GML
       (QCJ CAR BGQRGL AR Pnumber
       DPMK     Works_on, Employee
       U F CPC   ESSN=SSN   L B Lname = 'Smith');
```

Q a a

- Two reserved characters: % and _

Query 10: Retrieve all employees whose address is in Houston, Texas.

```
Q10:  QCJ CAR  *
      DPMK     Employee
      U F CPC  Address J GI C '%Houston,TX%';
```

Query 11: Retrieve all employees whose SSN has '88' at the end.

```
Q11:  QCJ CAR  *
      DPMK     Employee
      U F CPC  SSN J GI C '_ _ _ _ _ _ _ _ _88';
```

Q a a

- Standard arithmetic operators: +, -, *, /

Query 12: show the resulting salaries if every employee working on "ProductX" is given 10% raise

Q12:
```
QCJ CAR   Fname, Lname, 1.1*Salary   Q INC_SAL
DPMK      Employee, Works_on, Project
U F CPC   SSN = ESSN   LB PNO = Pnumber
           LB Pname = 'ProductX';
```

LSJJ a

AND	True	False	Unknown
True	T	F	U
False	F	F	F
Unknown	U	F	U

OR	True	False	Unknown
True	T	T	T
False	T	F	U
Unknown	T	U	U

NOT	
True	F
False	T
Unknown	U

QCJ CAR * DPMK Employee U F CPC SuperSSN IS NULL;

QCJ CAR * DPMK Employee U F CPC SuperSSN IS NOT NULL;

QCJ CAR A

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM TableName [alias] [, ...]
[WHERE condition]
[GROUP BY columnList]    [HAVING condition]
[ORDER BY columnList]
```

L O

- Complete SELECT-FROM-WHERE blocks within WHERE clause of another query
- Comparison operator IN
  - Compares value *v* with a set (or multiset) of values *V*
  - Evaluates to *TRUE* if *v* is one of the elements in *V*

Query 13: Retrieve the name and address of all employees who work for the 'Research' department

```
Q13:  QCJCAR  Fname, Lname, Address
      DPMK    Employee
      U F CPC Dno GL ( QCJCAR Dnumber
                       DPMK    Department
                       U F CPC Dname = 'Research' );
```

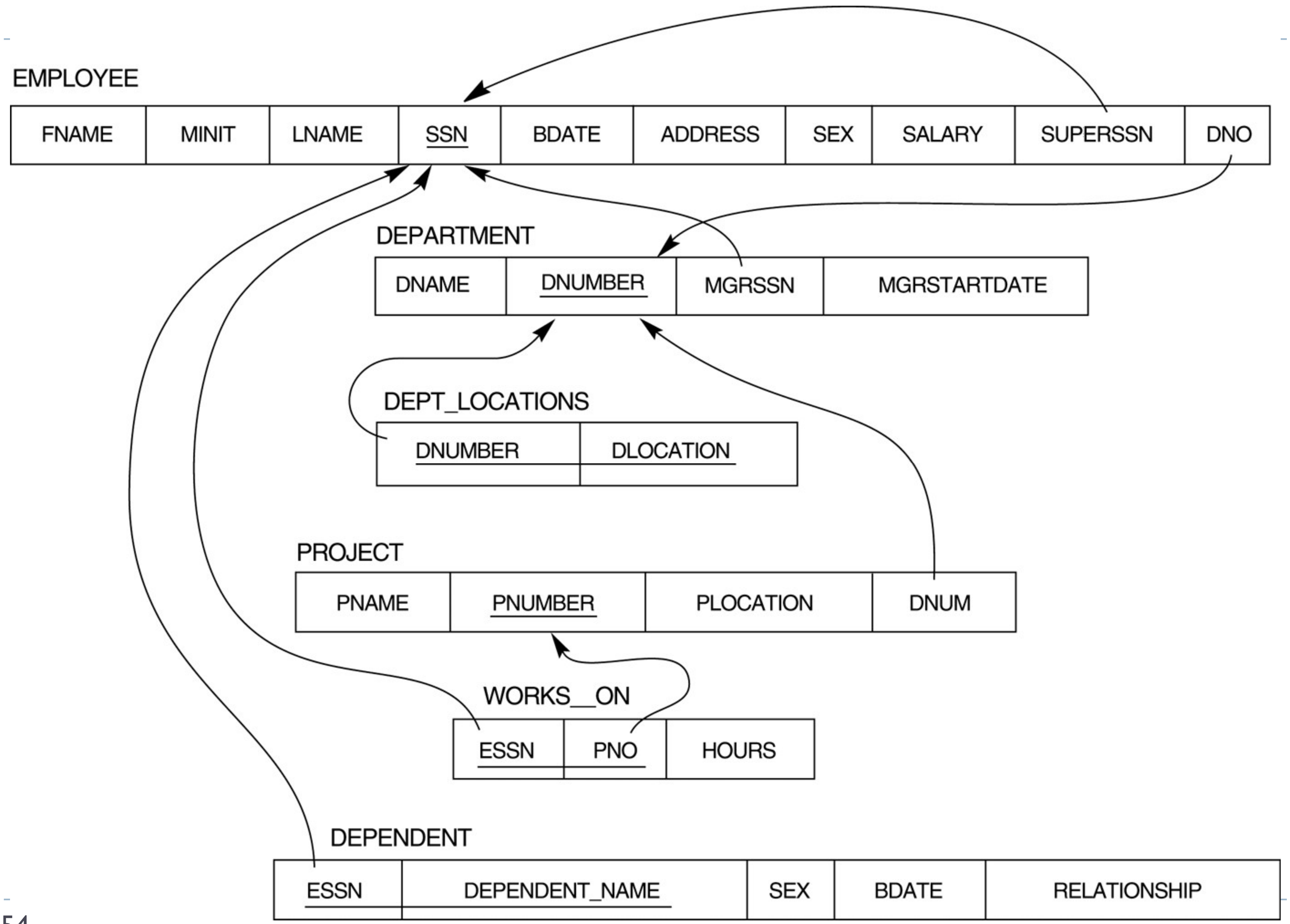
A L O .

- If a condition in the WHERE-clause of a nested query references an attribute of a relation declared in the outer query , the two queries are said to be **a**__________

Query 14: Retrieve the name of each employee who has a dependent with the same first name as the employee.

```
Q14:  QCJCAR   E.Fname, E.Lname
      DPMK     Employee E
      U F CPC  E.SSN GL ( QCJCAR  ESSN
                          DPMK    Dependent
                          U F CPC ESSN = E.SSN   LB
                              E.Fname = Dependent_name);
```

R AMK N L WB


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

A L O

- A query written with nested SELECT-FROM-WHERE blocks and using IN comparison operator can always be expressed as a single block query
- For example, Q14 may be written as in Q14A

```
Q14a: QCJ CAR   E.Fname, E.Lname
      DPMK      Employee E, Dependent D
      U F CPC   E.SSN = D.ESSN   L B
                E.Fname = D.Dependent_name;
```

L O C a

Query 15: Retrieve the SSNs of all employees who work the same (project, hours) combination on some project that employee John Smith (SSN=123456789) works on (using a nested query)

```
Q15:  QCJCAR   BGQRGLAR  ESSN
      DPMK     Works_on
      UFCPC    (PNO, Hours) GL
                          ( QCJCAR PNO, Hours
                            DPMK   Works_on
                            UFCPC  ESSN = '123456789' );
```

K A M

- Operators that can be combined with ANY (or SOME), ALL: =, >, >=, <, <=, and <>

Query 16: Retrieve all employees whose salary is greater than the salary of all employees in department 5

Q16:
```
QCJ CAR
DPMK      Employee
U F CPC   Salary >   J J ( QCJ CAR Salary
                           DPMK    Employee
                           U F CPC DNO=5 );
```

CVGQRQ SLGOSCD a

- **CVGQRQ** and **LMR CVGQRQ** function
  - Typically used in conjunction with a correlated nested query
  - EXISTS(Q) returns TRUE if the result of a query Q is NOT empty (Some tuples EXIST in the result).
  - NOT EXISTS(Q) returns TRUE if the result of a query Q is empty (No tuples are in the result).
- **SLGOSC O** function
  - Returns TRUE if there are no duplicate tuples in the result of query Q

CVQQRQD a

Query 14: Retrieve the name of each employee who has a dependent with the same first name as the employee

```
Q14b: QCJCAR  Fname, Lname
      DPMK    Employee
      UFCPC   CVQQRQ( QCJCAR *
                      DPMK   Dependent
                      UFCPC  ESSN = SSN   LB
                             FName = Dependent_name);
```

CVGQRQD a

Query 17: Retrieve the names of employees who have no dependents

```
Q17:  QCJCAR   Fname, Lname
      DPMK     Employee
      UFCPC    LMR CVGQRQ ( QCJCAR  *
                            DPMK   Dependent
                            UFCPC SSN = ESSN);
```

- In Q17, the correlated nested query retrieves all DEPENDENT tuples related to an EMPLOYEE tuple. If none exist , the EMPLOYEE tuple is selected

C Q

- An explicit (enumerated) set of values in the WHERE-clause

Query 18: Retrieve the SSNs of all employees who work on project numbers 1, 2, or 3.

Q18: **QCJCAR BCQRCL AR** ESSN
DPMK Works_on
U FCPC PNO **CL** (1, 2, 3);

H P

- Can specify a "joined relation" in the FROM-clause
- Allows the user to specify different types of joins
  - EQUIJOIN
  - NATURAL JOIN
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL OUTER JOIN

H R M H

- Joined table
  - Permits users to specify a table resulting from a join operation in the FROM clause of a query

Query 1: Retrieve the name and address of all employees who work for the 'Research' department.

Q1a: QCJ CAR Fname, Lname, Address
DPMK (Employee HMGL Department ML Dno = Dnumber)
U F CPC Dname = ‘Research’;

Q1: QCJ CAR Fname, Lname, Address
DPMK Employee, Department
U F CPC Dname='Research' LB Dnumber= Dno;

H R M H

- Specify different types of join
  - NATURAL JOIN
  - Various types of OUTER JOIN
- NATURAL JOIN on two relations R and S
  - **L a a**
  - Implicit EQUIJOIN condition for each pair of attributes with same name from R and S

H R QOJ M H a

- RIGHT OUTER JOIN
  - Every tuple in RIGHT table must appear in result
  - If no matching tuple
    - Padded with NULL values for the attributes of LEFT table
- FULL OUTER JOIN

H P C

Query 3: For each employee, retrieve the employee's name, and the name of his or her immediate supervisor.

```
Q3a:  QCJCAR  E.Fname, E.Lname, S.Fname, S.Lname
      DPMK    Employee E S
      UFCPC   E.SuperSSN = S.SSN;

Q3c:  QCJCAR  E.Fname, E.Lname, S.Fname, S.Lname
      DPMK    ( Employee E JCDR MS RCP HMGL
                Employee S ML  E.SuperSSN = S.SSN );
```

- **A**

H P C

Query 1: Retrieve the name and address of all employees who work for the 'Research' department.

Q1: QCJCAR Fname, Lname, Address
DPMK Employee, Department
UFCPC Dname = 'Research' LB Dnumber = Dno;

- could be written as:

Q1a: QCJCAR Fname, Lname, Address
DPMK (Employee **HMGL** Department **ML** Dnumber = Dno)
UFCPC Dname = 'Research';

Q1b: QCJCAR Fname, Lname, Address
DPMK (Employee **L RSP J HMGL** (Department
Q Dept(Dname, Dno, MSSN, MSDate)))
UFCPC Dname = 'Research';

H P C

Query 2: For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birthdate

Q2a: **QCJCAR** Pnumber, Dnum, Lname, Bdate, Address
DPMK ((Project **HMGL** Department **ML** Dnum = Dnumber) **HMGL** Employee **ML** MGRSSN = SSN))
U F CPC Plocation = 'Stafford’ ;

EEPCE RC DS L ARGML Q

- **AMS LR QS K K V KGL TE**

Query 19: Find the max, min, & average salary among all employees

Q19: **QCJCAR** **K V**(Salary), **KGL**(Salary), **TE**(Salary)
DPMK Employee;

EEPCE RC DS L ARGML Q

Queries 20: Retrieve the total number of employees in the company

Q20: **QCJ CAR AMS L R** (*)
DP MK Employee;

Queries 21: Retrieve the number of employees in the 'Research' department

Q21: **QCJ CAR AMS L R** (*)
DP MK Employee, Department
U F CPC Dno = Dnumber **L B** Dname = 'Research';

- Note: <u>NULL values are discarded </u>wrt. aggregate functions as applied to a particular column

EPMS NGLE

- A GROUP BY-clause is for specifying the grouping attributes, which must also appear in the SELECT-clause
- Each subgroup of tuples consists of the set of tuples that have the same value for the grouping attribute(s)
- Apply the aggregate functions to subgroups of tuples in a relation
- Each subgroup of tuples consists of the set of tuples that have the same value for the grouping attribute(s)
- The aggregate function is applied to each subgroup independently
- If NULLs exist in grouping attribute
  - Separate group created for all tuples with a NULL value in grouping attribute

QCJ CAR A

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM TableName [alias] [, ...]
[WHERE condition]
[GROUP BY columnList]    [HAVING condition]
[ORDER BY columnList]
```

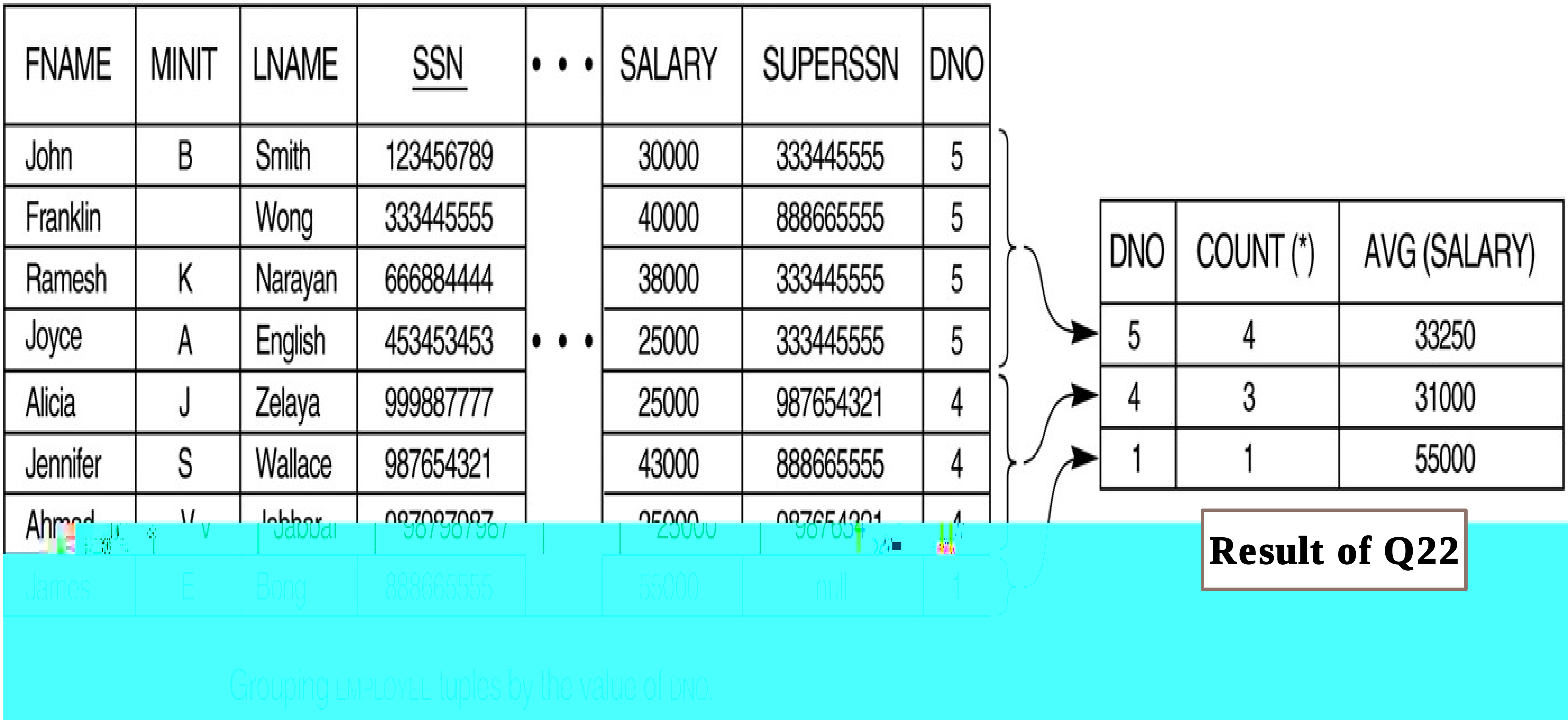
EPMS NGL E

Query 22: For each department, retrieve the department number, the number of employees in the department, and their average salary

Q22: **QCJ CAR** Dno, **AMS L R** (*), **TE** (Salary)
DPMK Employee
EPMS N BY Dno;

- In Q22, the EMPLOYEE tuples are divided into groups - each group having the same value for the grouping attribute DNO
- The COUNT and AVG functions are applied to each such group of tuples separately
- The SELECT-clause includes only the grouping attribute and the functions to be applied on each group of tuples
- A join condition can be used in conjunction with grouping

EPMSNGLE O



FNAME	MINIT	LNAME	SSN	• • •	SALARY	SUPERSSN	DNO
John	B	Smith	123456789		30000	333445555	5
Franklin		Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	• • •	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Borg	888665555		55000	null	1

DNO	COUNT (*)	AVG (SALARY)
5	4	33250
4	3	31000
1	1	55000

Result of Q22

Grouping EMPLOYEE tuples by the value of DNO.

EPMSNGLE RFCF TGLE AJ SQC

- Sometimes we want to retrieve the values of these functions for only those groups that satisfy certain conditions
- **R F TGLE a** is used for specifying **a a** (rather than on individual tuples)

EPMS NGL E RF C F TGL E AJ S QC

Query 23: For each project on which more than two employees work , retrieve the project number, project name, and the number of employees who work on that project.

```
Q23:  QCJ CAR     Pnumber, Pname, AMS L R (*)
      DP MK       Project, Works_on
      U F CPC     Pnumber = Pno
      EPMS N  W   Pnumber, Pname
      F  TGL E    AMS L R (*) > 2;
```

MPBCP W

- The ORDER BY clause is used to sort the tuples in a query result based on the values of some attribute(s)

Query 24: Retrieve a list of employees and the projects each works in, ordered by the employee's department, and within each department ordered alphabetically by employee last name

```
Q24:  QCJ CAR Dname, Lname, Fname, Pname
      DPMK    Department, Employee, Works_on, Project
      U F CPC Dnumber = Dno   LB SSN = ESSN
                         LB Pno = Pnumber
      MPBCP W Dname, Lname [DESC|ASC]
```

QCJ CAR A

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM TableName [alias] [, ...]
[WHERE condition]
[GROUP BY columnList]    [HAVING condition]
[ORDER BY columnList]
```

QCJ CAR A

QCJ CAR	Specifies which columns are to appear in output
DPMK	Specifies table(s) to be used
U F CPC	Filters rows
EPMSN W	Forms groups of rows with same column value
F TGLE	Filters groups subject to some condition
MPBCP W	Specifies the order of the output

A

1	Introduction of Structured Query Language
2	DDL: create, drop, alter
	BK J a
4	DCL: commit, rollback, grant, revoke

G A

- Add one or more tuples to a relation
- Attribute values should be listed in the same order as the attributes were specified in the CREATE TABLE command

INSERT INTO *TableName (Attribute1, Attribute2, ...)*
VALUES (*value1, value2, ...*);

G A

- Insert a tuple for a new EMPLOYEE:

U1: GL QCPR GL RM Employee
T JSCQ ('Richard', 'K', 'Marini', '653298653',
'30-DEC-52', '98 Oak Forest, Katy, TX',
'M', 37000, '987654321', 4);

- An alternate form of INSERT specifies explicitly the attribute names that correspond to the values in the new tuple, attributes with NULL values can be left out
- Example: Insert a tuple for a new EMPLOYEE for whom we only know the FNAME, LNAME, and SSN attributes.

U2: GL QCPR GL RM Employee (Fname, Lname, SSN)
T JSCQ ('Richard', 'Marini', '653298653');

G A

- Important note: Only the constraints specified in the DDL commands are automatically enforced by the DBMS when updates are applied to the database
- Another variation of INSERT allows insertion of multiple tuples resulting from a query into a relation

G A

- Example: *Suppose we want to create a temporary table that has the name, number of employees, and total salaries for each department. A table DEPTS_INFO is created by U3, and is loaded with the summary information retrieved from the database by the query in U3A*

```
U3:     APC  RCR   JC  Depts_info
                 ( Dept_name    VARCHAR(10),
                   No_of_emps   INTEGER,
                   Total_sal    INTEGER);

U3A:   GLQCPR GLRM Depts_info (Dept_name, No_of_emps,
                                          Total_sal)
         QCJCAR     Dname, AMSLR (*), QSK (Salary)
         DPMK       Department, Employee
         U F CPC    Dnumber = Dno
         EPMSN   W Dname;
```

B A

DELETE FROM *TableName*

WHERE *Condition*;

- Removes tuples from a relation
- Tuples are deleted from only one table at a time (unless CASCADE is specified on a referential integrity constraint)
- A missing WHERE-clause specifies that all tuples in the relation are to be deleted; the table then becomes an empty table
- The number of tuples deleted depends on the number of tuples in the relation that satisfy the WHERE-clause

B A C

```
U4A:  BCJ CRC DPMK     Employee
      U F CPC          Lname = 'Brown';

U4B:  BCJ CRC DPMK     Employee
      U F CPC          SSN = '123456789';

U4C:  BCJ CRC DPMK     Employee
      U F CPC          Dno GL
                 (QCJ CAR      Dnumber
                 DPMK          Department
                 U F CPC       Dname = 'Research');

U4D:  BCJ CRC DPMK     Employee;
```

S A

```
UPDATE TableName
SET    Set-Clause
WHERE  Condition;
```

- Used to modify attribute values of one or more selected tuples
- A WHERE-clause selects the tuples to be modified
- An additional SET-clause specifies the attributes to be modified and their new values
- Each command modifies tuples in the same relation
- Referential integrity should be enforced

S A

- Example: *Change the location and controlling department number of project number 10 to 'Bellaire' and 5, respectively.*

```
U5:   S NB  RC      Project
      QCR           Plocation = 'Bellaire', Dnum = 5
      U F CPC       Pnumber = 10;
```

S A

- Example: *Give all employees in the 'Research' department a 10% raise in salary.*

```
U6:    S NB  RC   Employee
       QCR        Salary = Salary *1.1
       U F CPC    Dno  GL (QCJ CAR  Dnumber
                            DPMK  Department
                            U F CPC Dname = 'Research');
```

a BBJ R

- ASSERTIONs to express constraints that do not fit in the basic SQL categories
- Mechanism: CREATE ASSERTION
  - components include: a constraint name, followed by CHECK, followed by a condition

a BBJ R

- Example: *The salary of an employee must not be greater than the salary of the manager of the department that the employee works for'*

```
APC RC QQCPRGML Salary_constraint
AFCAI (LMR CVGQRQ (QCJ CAR *
                    DPMK Employee E, Employee M,
                         Department D
                    U F CPC E.Salary > M.Salary  L B
                        E.Dno = D.Number  L B
                        D.MGRSSN = M.SSN));
```

a BBJ R

- Triggers: to specify the type of action to be taken as certain events occur and as certain conditions are satisfied
- Details of triggers: presentation and lab

T

- A view is a “virtual” table that is derived from other tables
- Allows for limited update operations (since the table may not physically be stored)
- Allows full query operations
- A convenience for expressing certain operations

TCCU

- Specify a different WORKS_ON table (view)

```
APC  RC TCCU        Works_on_new   Q
     QCJ CAR        Fname, Lname, Pname, Hours
     DPMK           Employee, Project, Works_on
     U F CPC        SSN = ESSN   LB Pno = Pnumber;
```

- We can specify SQL queries on a newly create table (view):

```
QCJ CAR Fname, Lname From Works_on_new
U F CPC Pname = ‘Seena’;
```

- When no longer needed, a view can be dropped:

```
BPMN TCCU  Works_on_new;
```

T S G T

- Update on a view defined on a single table without any aggregate functions
  - Can be mapped to an update on underlying base table
- View involving joins
  - Often not possible for DBMS to determine which of the updates is intended

A

1	Introduction of Structured Query Language
2	DDL: create, drop, alter
3	DML: select, insert, update, delete
	B AJ a a **A**

QOJ B A

- Commands:
  - GRANT
  - REVOKE
- Based on three central objects:
  - Users
  - Database objects
  - Privileges: select, modify (insert, update, delete), reference

QOJ B A

- GRANT: pass privileges on their own database objects to other users

```
GRANT     <privilege list>
ON        <database objects>
TO        <user list>
```

- REVOKE: take back (cancel) privileges on their own database objects from other users

```
REVOKE    <privilege list>
ON        <database objects>
FROM <user list>
```

QOJ B A

- Propagation of Privileges using the GRANT OPTION
  - Whenever the owner A of a relation R grants a privilege on R to another account B, privilege can be given to B with or without the GRANT OPTION.
  - If the GRANT OPTION is given, this means that B can also grant that privilege on R to other accounts.

C

- Suppose that the DBA creates four accounts
  - A1, A2, A3, A4
- and wants only A1 to be able to create base relations. Then the DBA must issue the following GRANT command in SQL

```
GRANT CREATETAB TO A1;
```

- In SQL2 the same effect can be accomplished by having the DBA issue a **APC RC QAF CK** command as follows:

```
CREATE SCHEMA EXAMPLE AUTHORIZATION
A1;
```

C .

- User account A1 can create tables under the schema called **CV K NJ C**.
- Suppose that A1 **a** the two base relations **CK NJ MWCC** and **BCN PRK CL R**
  - A1 is then of these two relations and hence all the relation privileges on each of them.
- Suppose that A1 wants to grant A2 the privilege to insert and delete tuples in both of these relations, but A1 does not want A2 to be able to propagate these privileges to additional accounts:

```
GRANT INSERT, DELETE ON
    EMPLOYEE, DEPARTMENT TO A2;
```

C

EMPLOYEE

Name	Ssn	Bdate	Address	Sex	Salary	Dno

DEPARTMENT

Dnumber	Dname	Mgr_ssn

Figure 23.1
Schemas for the two relations EMPLOYEE and DEPARTMENT.

C

- Suppose that A1 wants to allow A3 to retrieve information from either of the two tables and also to be able to propagate the SELECT privilege to other accounts.
- A1 can issue the command:

```
GRANT SELECT ON EMPLOYEE, DEPARTMENT
    TO A3 WITH GRANT OPTION;
```

- A3 can grant the **QCJ CAR** privilege on the **CK NJ MWCC** relation to A4 by issuing:

```
GRANT SELECT ON EMPLOYEE TO A4;
```

  - Notice that A4 can’t propagate the SELECT privilege because GRANT OPTION was not given to A4

C

- Suppose that A1 decides to revoke the SELECT privilege on the EMPLOYEE relation from A3; A1 can issue:

```
REVOKE SELECT ON EMPLOYEE FROM A3;
```

- The DBMS must now automatically revoke the SELECT privilege on EMPLOYEE from A4, too, because A3 granted that privilege to A4 and A3 does not have the privilege any more.

C

- Suppose that A1 wants to give back to A3 a limited capability to SELECT from the EMPLOYEE relation and wants to allow A3 to be able to propagate the privilege.
  - The limitation is to retrieve only the NAME, BDATE, and ADDRESS attributes and only for the tuples with DNO=5.
- A1 then create the view:

```
CREATE VIEW A3EMPLOYEE AS
   SELECT NAME, BDATE, ADDRESS
   FROM EMPLOYEE
   WHERE DNO = 5;
```

- After the view is created, A1 can grant **QCJ CAR** on the view A3EMPLOYEE to A3 as follows:

```
GRANT SELECT ON A3EMPLOYEE TO A3
      WITH GRANT OPTION;
```

C

- Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE;
- A1 can issue:

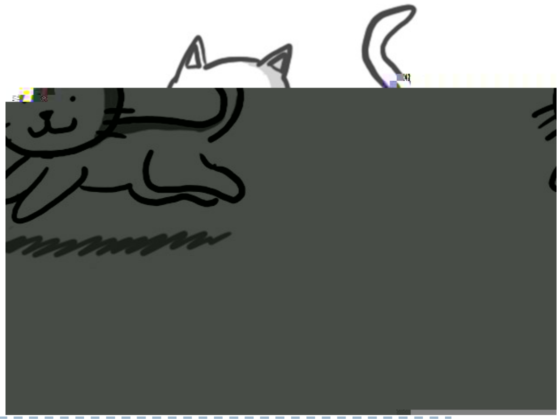
```
GRANT UPDATE ON EMPLOYEE (SALARY) TO
A4;
```

- The UPDATE or INSERT privilege can specify particular attributes that may be updated or inserted in a relation.
  - Other privileges (SELECT, DELETE) are not attribute specific.

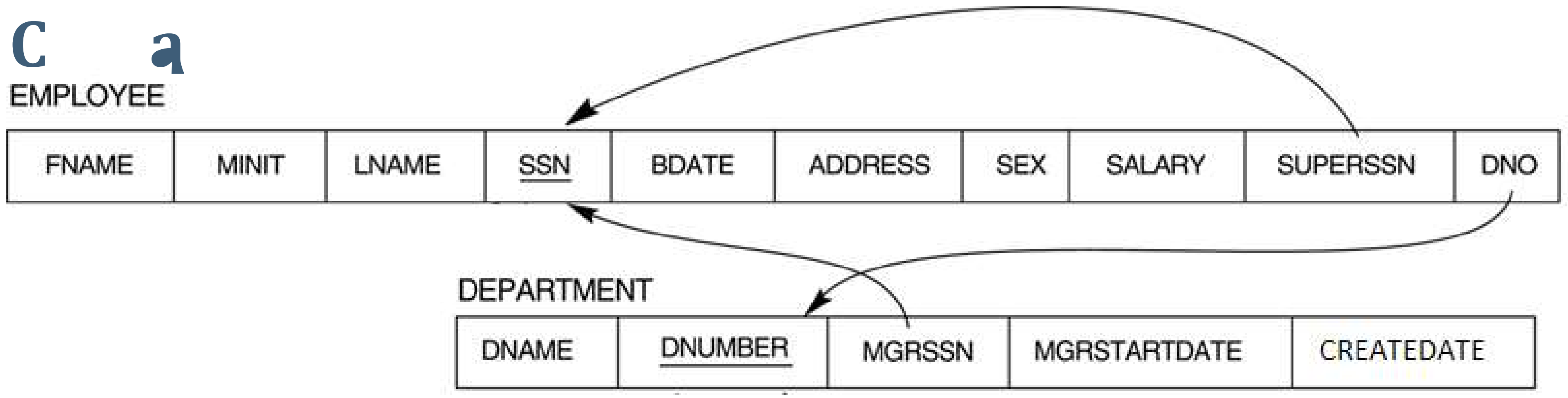
Q

- SQL developments: an overview
- SQL
  - DDL: Create, Alter, Drop
  - DML: select, insert, update, delete
  - Introduction to advanced DDL (assertions & triggers), views, DCL (commit, rollback, grant, revoke)

Q
A



C a



CK NJ MWCC

- Fname, Lname: VARCHAR(15), NOT NULL
- Minit: CHAR
- SSN: CHAR(9), NOT NULL, PRIMARY KEY
- Bdate: DATE, <= “1/1/1999”
- Address: VARCHAR(100)
- Sex: CHAR, {F/M}
- Salary: DECIMAL(10,2)
- SuperSSN: CHAR(9), refers to EMPLOYEE(SSN)
- Dno: INT, NOT NULL, default value = 1, refers to DEPARTMENT(Dnumber) – ON DELETE SET DEFAULT

BCN PRK CLR

- Dname: VARCHAR(15), NOT NULL, UNIQUE
- Dnumber: INT, NOT NULL, PRIMARY KEY
- MgrSSN: CHAR(9), NOT NULL, default value = ‘888665555’, refers to EMPLOYEE(SSN) – ON DELETE SET DEFAULT, ON UPDATE CASCADE
- MgrStartDate: DATE
- CreateDate: DATE, <= MgrStartDate

EMPLOYEE

Fname	Minit	Lname	SSN	Bdate	Address	Sex	Salary	SuperSSN	DNO
An	V	Nguyen	123456789	1/1/1990	TP.HCM	F	10,000	987654321	1
Binh	T	Nguyen	987654321	2/2/1988	Ha noi	M	15,000		1
Hoa	T	Tran	111222333	3/3/1991	Binh Duong	F	12,000	123456789	2
Long	K	Ly	888665555	4/4/1993	Dong Nai	M	20,000	987654321	2

DEPARTMENT

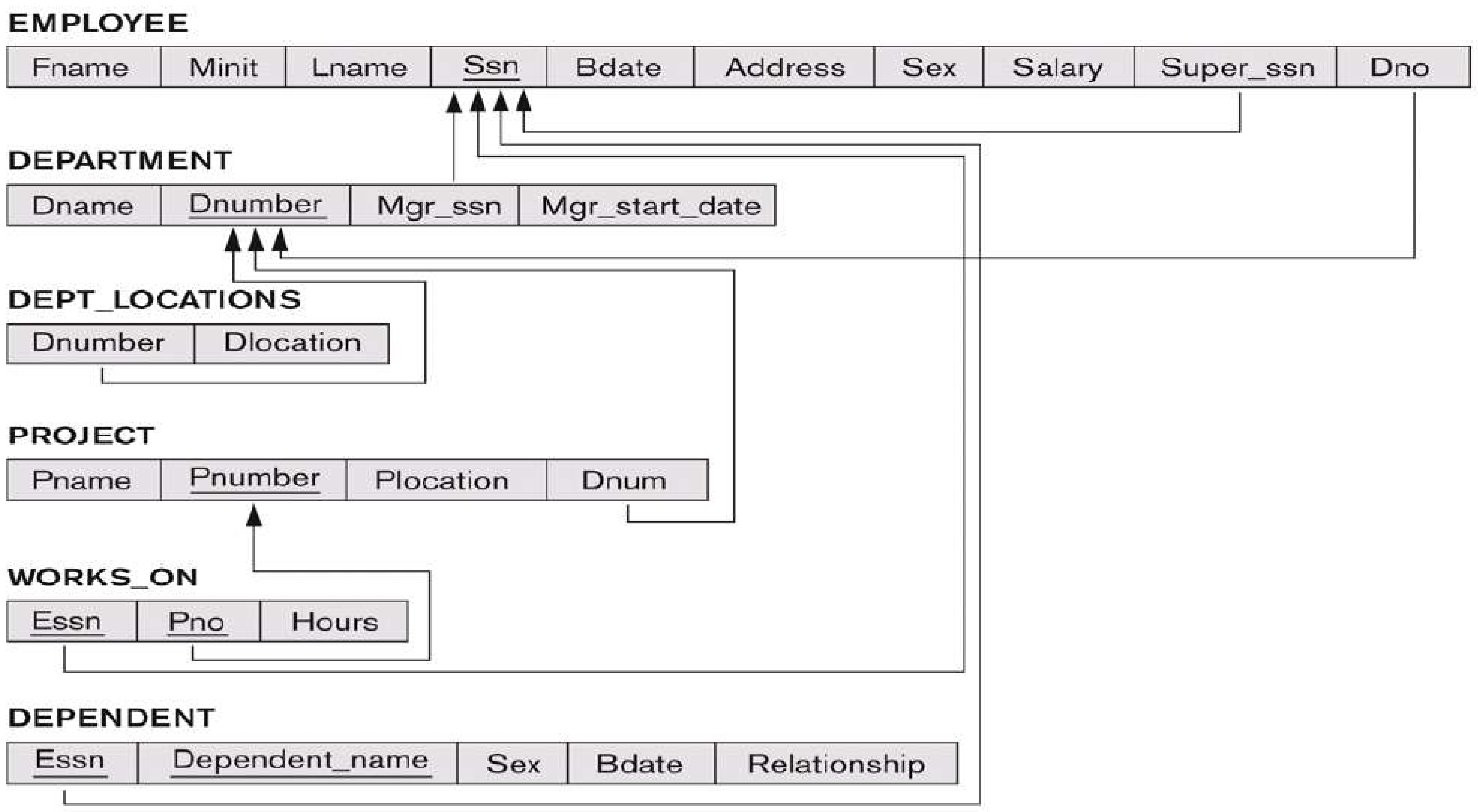
Dname	Dnumber	MgrSSN	MgrStartDate	CreateDate
NH	1	123456789	10/10/2010	1/1/1999
TC	2	888665555	5/5/2000	1/1/1999

APC RC R J C

```
CREATE TABLE [SchemaName.]TableName
({colName dataType [NOT NULL] [UNIQUE] [PRIMARY KEY]
[DEFAULT defaultOption]
[CHECK searchCondition] [,...]}
[PRIMARY KEY (listOfColumns),]
{[UNIQUE (listOfColumns),] [...,]}
{[FOREIGN KEY (listOfFKColumns)
 REFERENCES ParentTableName [(listOfCKColumns)]
 [ON UPDATE referentialAction]
 [ON DELETE referentialAction ]] [,...]}
{[CHECK (searchCondition)] [,...] })
```

APC RC BCN PRK CL R

```
Dname VARCHAR(15)  NOT NULL  UNIQUE,
Dnumber INT  NOT NULL PRIMARY KEY,
MgrSSN: CHAR(9)  NOT NULL  DEFAULT
‘888665555’,
MgrStartDate DATE,
CreateDate  DATE,
CHECK (CreatDate <= MgrStartDate)
```


EMPLOYEE
Fname
Minit
Lname
Ssn
Bdate
Address
Sex
Salary
Super_ssn
Dno
DEPARTMENT
Dname
Dnumber
Mgr_ssn
Mgr_start_date
DEPT_LOCATIONS
Dnumber
Dlocation
PROJECT
Pname
Pnumber
Plocation
Dnum
WORKS_ON
Essn
Pno
Hours
DEPENDENT
Essn
Dependent_name
Sex
Bdate
Relationship

C a

1. Retrieve the names of all employees in the departments which are located in Houston
2. List the names of all employees who have a dependent with the same first name as themselves
3. For each project, calculate the total number of employees who work for it, and the total number of hours that these employees work for the project.
4. Retrieve the average salary of all female employees.
5. For each department whose average employee salary is more than $30.000, retrieve the department name and the number of employees work for that department.

C a

1. Retrieve the name and address of all employees who work for the department which is managed by John B. Smith.
2. For every project located in ‘Stafford’, list the project number, the controlling department name, and the number of employees working on it (project).
3. Find the names of employees who work on all the projects controlled by department number 5.
4. List the names of all employees with more than 2 dependents.
5. Retrieve the names of employees who have no sons as dependents.
6. Retrieve the average salary of all female employees.
7. For each department whose average employee salary is more than $30.000, retrieve the department name and the number of employees work for that department.