University of Ottawa
CSI 2132 and 2532 – Final Examination
Professor(s): Herna L. Viktor and Iluju Kiringa

21 April 2012
09:h30-12h30
Duration: 3 hrs

Memo of Questions Set by HLV

Family name: _____

First name: _____

Student number: _____

There are 6 questions and a total of 100 points.

This exam must contain 16 pages, including this cover page.

1 – EER Diagram	**/ 15**
2 – Relational Algebra and Relational Calculus	**/ 15**
3 – The Relational Model and SQL	**/ 15**
4 – Normal Forms	**/ 20**
5 – Physical database design	**/ 15**
6 – Storage and Indexing	**/ 20**
Total	**/ 100**

2 Relational Algebra and Relational Calculus — 15 points

Consider the following relational schema about the Nursing staff working in Hospitals across rural Ontario.

$Nurse(\underline{nid} : \texttt{int}, nname : \texttt{string}, age : \texttt{real}, salary : \texttt{real}, sid : \texttt{int})$
$Supervisor(\underline{sid} : \texttt{int}, rating : \texttt{real})$
$WorksIn(\underline{nid} : \texttt{int}, \underline{hid} : \texttt{int}, hours : \texttt{real})$
$Hospital(\underline{hid} : \texttt{int}, hname : \texttt{string}, tid : \texttt{int})$
$Town(\underline{tid} : \texttt{int}, tname : \texttt{string}, mayor : \texttt{string})$

Part A — 4 points Provide the relational algebra expression to display the names and salaries of all nurses who do not work at any hospital.

```
(Note: We will use S for SELECT, P for PROJECT, * for NATURAL JOIN,
- for SET DIFFERENCE)

 ALL_Nurses <-- Project Nid (Nurse)
 WORKING_Nurses(Nid) <-- P Nid (WorksIn)
 NON_WORKING_Nurses <-- ALL_Nurses - WORKING_Nurses
 RESULT <-- P LNAME,FNAME,Salary (Nurse * NON_WORKING_Nurses)}
```

Part B — 5 points Provide the relational algebra expression to display the names of all nurses, younger than 30 years old, who worked at all the hospitals that are based in a town called Campbellford.

```
(Note: We use -:- to denote the DIVISION operation *)

 Hospital_Nurses(Hid, Nid)<-- P Hid, Nid (WorksIn)
 ALL_KingsHospitals(Hid) <-- P Hid (S(Tname = Kingston) Town)
 EMP_ALL_KingsHospitals <-- PROJ_Hospital_Nurses -:- ALL_KingsHospitals
 RESULT <-- P LNAME,FNAME ( S (age < 30) Nurses * EMP_ALL_KingsHospitals)
```

... Cont'd

Part C — 3 points Explain, by means of your own example, the main difference between tuple and domain relational calculus.

Tuple relational calculus: Variables range over (is bound to) tuples

Domain relational calculus: Variables range over (is bound to) domain variables

Example tuple:

```
{ N | N (ELEMENT OF) Nurse (AND) N.age < 30}
```

Example domain:

```
{<I, N, A, M, S> | {<I, N, A, M, S> (ELEMENT OF) Nurse (AND) A < 30}
```

Part D — 3 points Explain, by means of your own example, what a safe relational calculus expression ensures.

Safe relational calculus ensures that a relational calculus expression has a finite set of answers, i.e. a finite set of tuples are returned and all values returned are from the domain of expression. Safe relational calculus is said to be relationally complete and is equivalent to relational algebra.

For example, an unsafe relational calculus expression would find all Nurses that are NOT in the Nurse relation; leading to an infinite answer set.

... Cont'd

4 Normal Forms — 20 points

Consider the following data recorded about contestants in the new **BigTeeth** TV reality show. This show focuses on conflicts between humans and animals with big teeth, such as crocodiles, tigers and lions. You are asked to run a background check on the contestants to determine whether a person has a police record (PR), to double-check his or her employment history (EMP), and so on. You create a table, which records the following information.

Name	YofB	PRecord	PRDate	PRCat	PROutcome	EmpName	EmpComments
John Smith	1980	Ottawa	12 Sept 11	Serious	Jail	JollyBouncer	Fired
John Smith	1980	Ottawa	1 Sept 08	Minor	Fine	JollyBouncer	Fired
John Smith	1979	none	none	none	none	Walmart	Good Employee
John Smith	1979	Toronto	1 Dec 06	Minor	Fine	none	none
Ann Doe	1975	none	none	none	none	Sears	Resigned
Ann Doe	1975	Montreal	3 Dec 08	Serious	Jail	none	none
Sam Watson	1981	none	none	none	none	none	none

A. (8 points) This table may be susceptible to a number of data inconsistencies, due to information redundancy. Explain, by means of your own examples with reference to the **BigTeeth** data, what potential problems (or anomalies) may occur.

This question tests insight into the subject matter. This table is full of redundancies and it is very difficult to identify persons. This is because there are no key constraints and no unique identifiers.

The answer should address any four of the following, together with examples:

Deletion anomalies: If we remove the information of say Ann Doe, the second tuple, then we lose the information about her priori police record.

Insertion anomalies: To insert a new candidate we need to include information about the employment and the police records, even if there are no details.

Modification anomalies: It is unclear whether the third and fourth tuples refer to the same person as the first two tuples. So, it is not clear if this is another person or an error in the Year of Birth.

Redundant storage: In this case, we potentially store information about the same person, more than once. This leads to wasted space.

Handling nulls: In this case study, it is not possible to infer whether a null means a) unknown, b) does not apply or c) known but absent. For example, does the absence of a police record imply no police record, or does it mean that this has not been verified?

... Cont'd

B. (4 points) Identify all the functional dependencies, as based on the data in the **BigTeeth** table.
We rename the attributes to ABCDEFGH

The following FDs may be inferred, based on this data and the assumption that *none* **means** *not recorded***.**

$$ABCG \rightarrow ABCDEFGH, E \rightarrow F, G \rightarrow H, CD \rightarrow E$$

C. (4 points) Normalize the relation to Boyce Codd Normal Form and show your resultant relations, indicating all primary and foreign keys.
A relation is in 1NF if all rows are uniquely identifiable. This is the case here, even though the choice of primary key is not that great. A better design would have been to create primary keys (numbers) for candidate, police record and employee.

The relation is in 2NF, since there are no partial dependencies.

The relation is not in 3NF, since there are a number of transitive dependencies. A solution is ABCG, CDE, EF and GH.

The decomposition above is in BCNF, since there are no cyclic references and all attributes are dependent on the key.

Please note that it is also possible to introduce your own keys as a first step.

D. (2 points) Determine whether your decomposition is dependency preserving and motivate your answer.

The answer will be marked considering your answer above.

The above decomposition is dependency preserving, since all dependencies may be checked by considering only one table of the decomposition.

E. (2 points) Determine whether your decomposition is a lossless-join decomposition and motivate your answer.
The answer will be marked considering your answer above.

This decomposition is a lossless-join decomposition, since the joining of the tables will not introduce extra tuples into the result.

... Cont'd

5 Physical database design — 15 points

Consider, for the last time, the following relational schema about the Nursing staff working in Hospitals across rural Ontario.

$Nurse(\underline{nid} : \texttt{int}, nname : \texttt{string}, age : \texttt{real}, salary : \texttt{real}, sid : \texttt{int})$
$Supervisor(\underline{sid} : \texttt{int}, rating : \texttt{real})$
$WorksIn(\underline{nid} : \texttt{int}, \underline{hid} : \texttt{int}, hours : \texttt{real})$
$Hospital(\underline{hid} : \texttt{int}, hname : \texttt{string}, tid : \texttt{int})$
$Town(\underline{tid} : \texttt{int}, tname : \texttt{string}, mayor : \texttt{string})$

Assume that there are currently no indexes defined on any of the tables. The Nurse relation spans 100 disk pages and the current buffer size is 10. Further, the WorksIn relation spans 20 disk pages. Consider the following query 1.

```
SELECT N.nname, W.hours
FROM Nurse N, WorksIn W
WHERE N.salary > 20000
AND N.nid = W.nid
AND hours = 100;
```

A. (5 points) Discuss a scenario where adding indexes may improve the efficiency of the query. Motivate your answer by listing two *different* indexing options to consider.

This question tests insight.

The issue to bring is that pushing up selects may speed up the query. Two possible solutions will be to put indexes on a) hours and c) salary. The success will depend on the distribution of the data, the workload, and the frequency of this query and the frequency of updates. Another potential index could be on R.nid and W.nid, to try to speed up the join.

... Cont'd

B. (4 points) Explain how the buffer manager processes a read request for a page.

The buffer manager will first determine whether page X is already in the buffer. If so, there is no need to read it again.

Further, if page X is already in the buffer, we need to determine whether it is dirty or not. The way that we handle dirty pages will depend on our concurrency control mechanism.

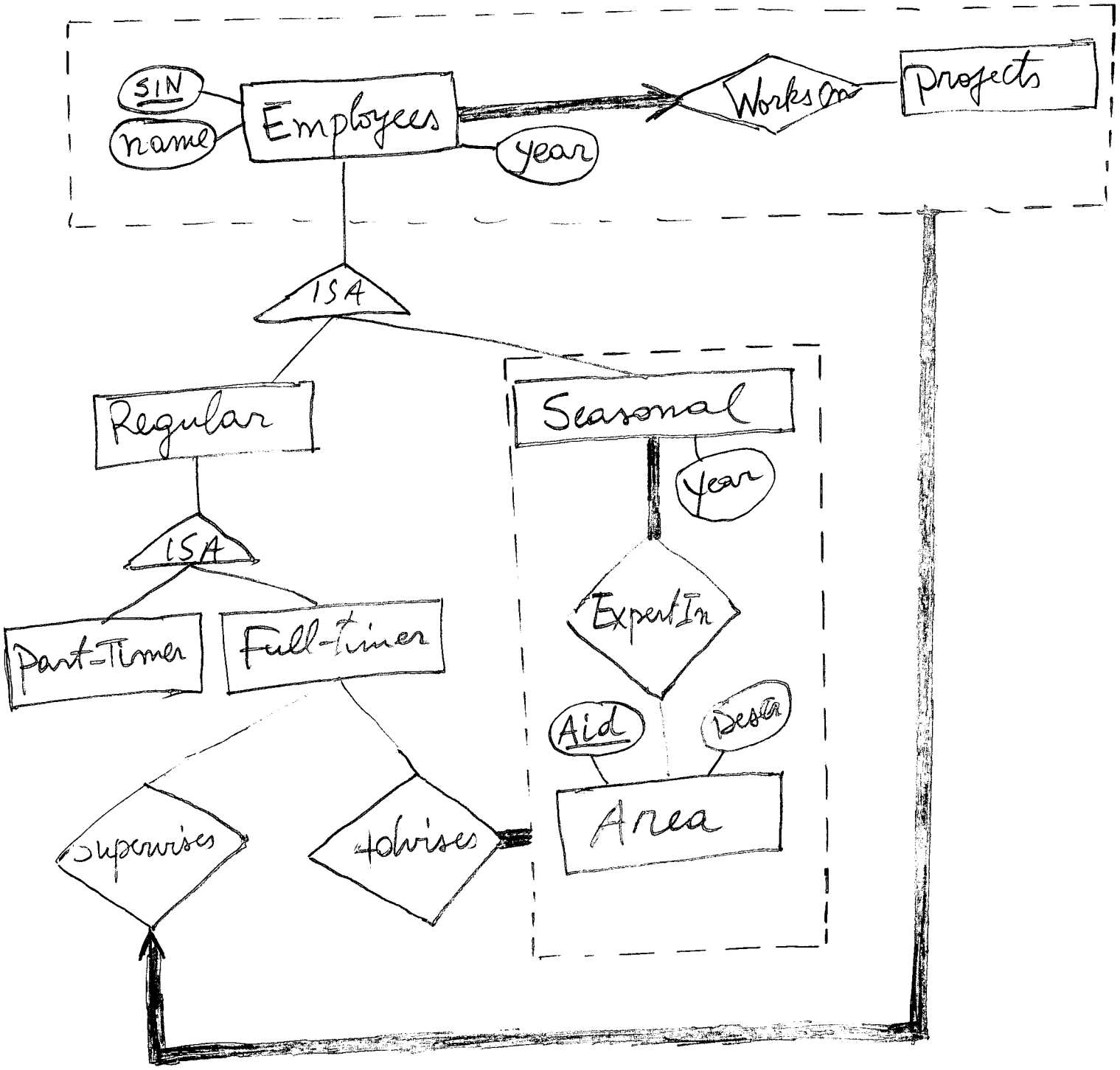
If page X is not in the buffer, a read request is sent to the disk space manager, with the address of page X on disk. Page X is then read into the buffer, provided that the buffer is not full. If the buffer is full, the buffer manager chooses another page Y for replacement. This page Y is written to the disk, and the buffer manager can proceed to read our page X after that.

C. (6 points) Having variable-length fields in a record may raise some subtle issues, especially when a record is modified. Discuss three (3) issues that may arise.

See p.332 of the text book; answer to be updated by HLV.

... Cont'd

Dessinez votre diagramme ER ci-bas.



... suite

3 Le Modèle Relationnel et SQL — 15 points

Considérez à nouveau le schéma relationel suivant de la question précédente.

Nurse(*nid* : int, *nname* : string, *age* : real, *salary* : real, *sid* : int)
Supervisor(*sid* : int, *rating* : real)
WorksIn(*nid* : int, *hid* : int, *hours* : real)
Hospital(*hid* : int, *hname* : string, *tid* : int)
Town(*tid* : int, *tname* : string, *mayor* : string)

Partie A — 3 points Explication d'une Requête

Considérez la requête suivante et expliquez ce qu'elle fait.

```
SELECT nname
FROM Nurse N
WHERE NOT EXISTS
    (  (SELECT    H.hid
        FROM      Hospital H
        WHERE     H.town = 'Toronto')
     EXCEPT       H.hname = "Montfort"
       (SELECT    W.hid
        FROM      WorksIn W
        WHERE     N.nid = W.nid))
```

Find the names of nurses who work in all hospitals named "Montfort"

... suite

Partie B — 4 points

Expliquez comment les vues peuvent être utilisées pour assurer l'indépendance des données.

By keeping an external data level separated from a logical data level.
Views restrict user access to a well defined subset of the logical level and are defined only in terms of the external data level.

Partie C — 8 points Requêtes SQL

1. Ecrivez la requête suivante en SQL:

"Pour chaque hôpital dans lequel travaillent plus de 100 infirmières, trouvez le nom dudit hôpital ainsi que le nombre d'infirmières qui y travaillent." (4 points)

```
SELECT  hname, COUNT(*)
FROM    Hospital H, WorksIn W
WHERE   H.hid = W.hid
GROUP BY  hname
HAVING    COUNT(*) > 100
```

... suite

2. Donnez les contraintes d'intégrité en SQL (une Clé primaire ou étrangère, un check, une assertion ou autre) pour traduire les exigences suivantes:

"Chaque superviseuse doit aussi être une infirmière." (2 points)

Use this check constraint:

```
CHECK (NOT EXISTS((SELECT sid FROM Supervisor)
                  EXCEPT
                  (SELECT nid FROM Nurse)))
```

Alternative solution: Use the following constraint in the table Supervisor:

```
FOREIGN KEY sid REFERENCES Nurse(nid)
```

"Chaque infirmière. doit avoir une superviseuse." (2 points)

Use a NOT NULL constraint as follows:

```
CREATE TABLE Nurse (
    ⋮
    sid  int NOT NULL
    ⋮
)
```

... suite

6 Indexes — 20 points

Partie A — 10 points Arbre B+

Considérez l'instance suivante de la relation *Sailors* vue en classe:

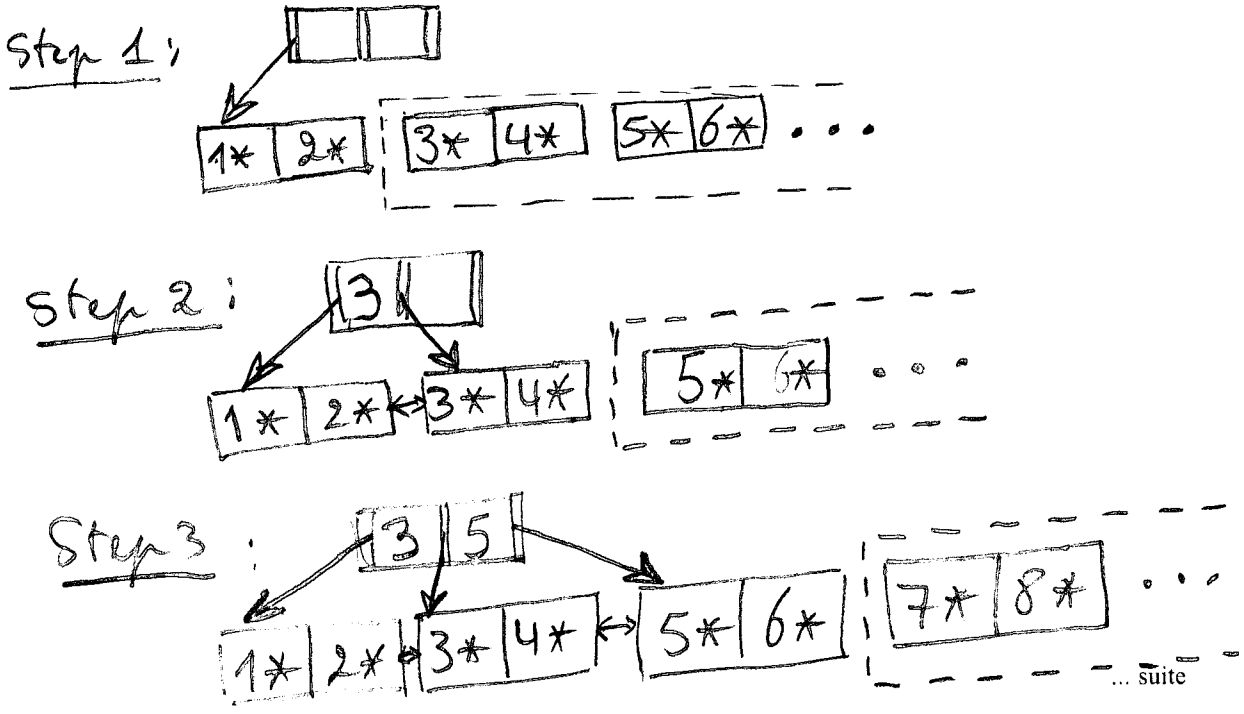
sid	sname	srating	age
7	'Sam'	10.0	67
1	'Joe'	2.0	71
3	'Sumo'	3.0	23
5	'Mikey'	10.0	67
4	'Jil'	2.0	71
2	'Noam'	3.0	23
6	'Ann'	10.0	67
8	'Louis'	2.0	71
10	'Nathan'	3.0	23
9	'Liu'	10.0	67
12	'Wang'	2.0	71
11	'Jones'	3.0	23

Dans cette question, vous devez utiliser l'algorithme de chargement en vrac ("bulk loading") pour construire un index à arbre B+ sur l'instance de la relation ci-haut. Utilisez l'alternative 1 vue en classe selon laquelle une entré de donnée $k*$ contient la donnée elle-même. Supposez que l'**ordre** de l'arbre est 1.

Convention : Right pointer of k points to values $\geq k$.

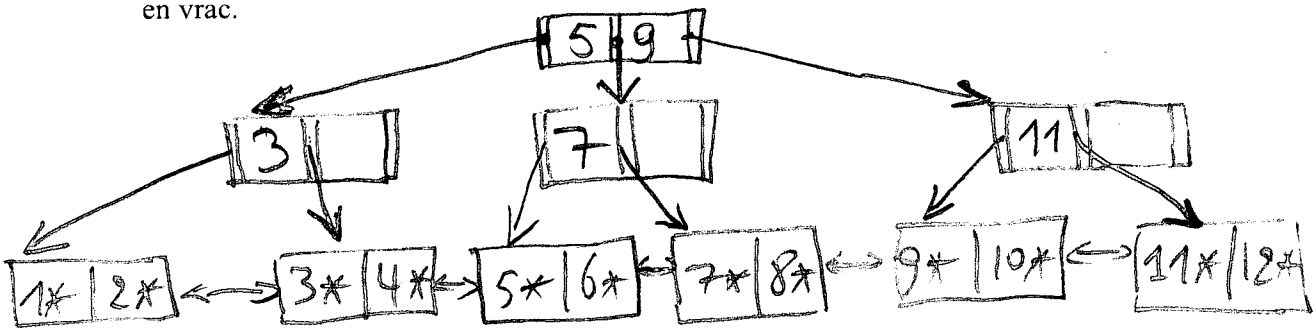
Répondez aux questions suivantes:

1. (4 Points) Montrez *seulement* les trois premiers pas du chargement en vrac.

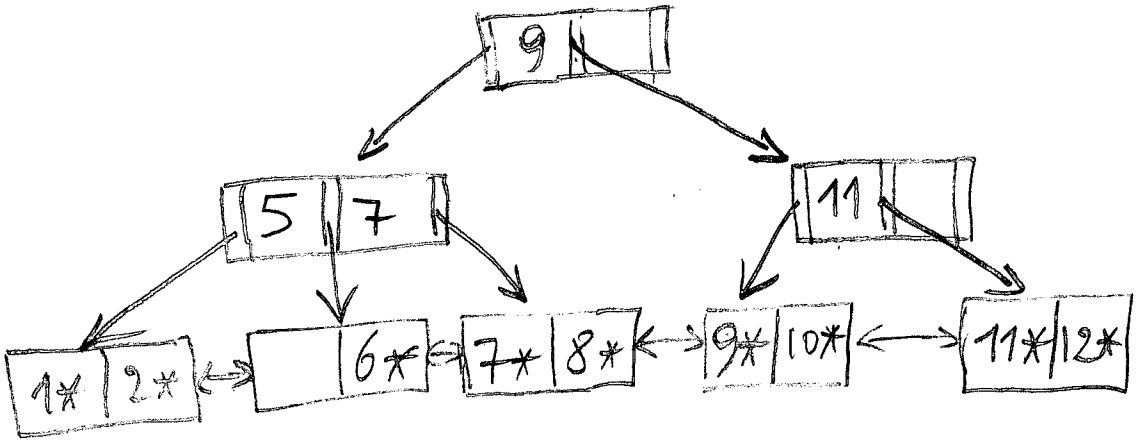


... suite

2. (2 Points) Montrez l'arbre final après l'exécution complète de l'algorithme du chargement en vrac.

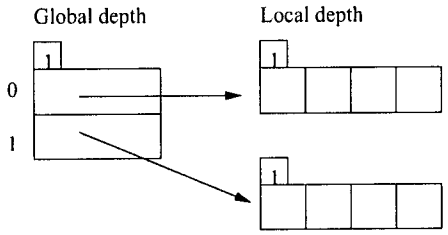


3. (4 points) Considérez l'arbre obtenu à l'étape (2) ci-dessus et montrez le après l'effacement (dans l'ordre donné) des tuples :
   $\langle 3,' Sumo', 3.0, 23\rangle$, $\langle 5,' Mikey', 10.0, 67\rangle$, $\langle 2,' Noam', 3.0, 23\rangle$, and $\langle 4,' Jil', 2.0, 71\rangle$.



... suite

Considérez la table à hachage initiale ci-bas:



Montrez les pas de l'algorithme du hachage extensible en insérant les clés suivantes (dans cet ordre!): $D*, H*, K*, M*, A*, O*, P*, R*, Y*, W*$ et $V*$. (Suggestion: suivez la méthode utilisée dans le manuel en ne dessinant une nouvelle figure que si la taille du répertoire doit être doublée.)

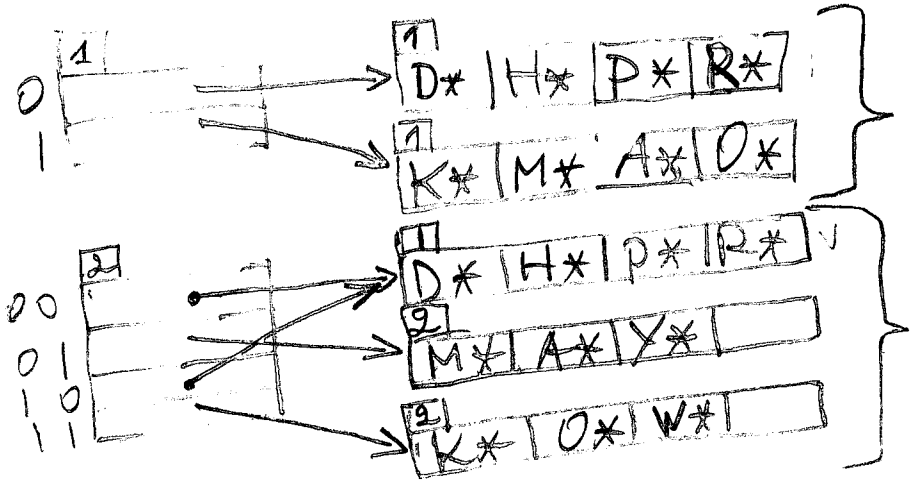


Table after inserting D*, H*, K*, M*, A*, O*, P*, R*.

Table after inserting the above, plus Y* and W*

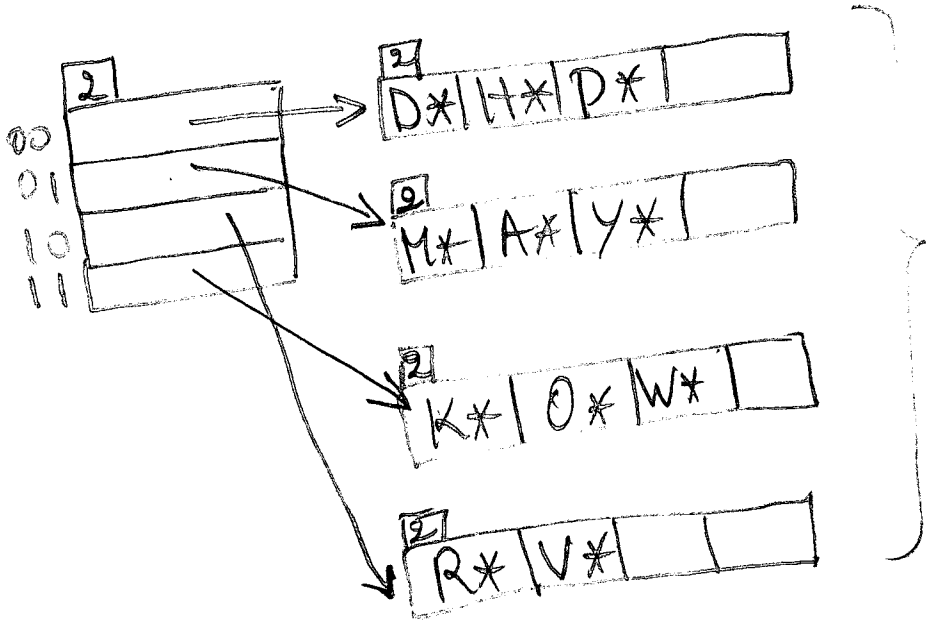


Table after inserting the above, plus V*.

... Fin de l'Examen Final