

Project: DRA FRONT END FILTER PROJECT

Title: Proof of Security (IIa)

Ref: DS/FMU/FEF/010 *Issue: Revision :* 2.5 *Date:* 5 December 2009

Status: Approved *Type:* Specification

Keywords:

Author:

<i>Name</i>	<i>Location</i>	<i>Signature</i>	<i>Date</i>
G. M. Prout	WIN01		

Authorisation for Issue:

<i>Name</i>	<i>Function</i>	<i>Signature</i>	<i>Date</i>
R.B. Jones	HAT Manager		

Abstract: This document provides a formal proof for the security property on *hide* for the DRA front end filter project RSRE 1C/6130.

Distribution: HAT FEF File
Simon Wiseman

0 DOCUMENT CONTROL

0.1 Contents List

0	DOCUMENT CONTROL	2
0.1	Contents List	2
0.2	Document Cross References	3
0.3	Changes History	3
0.4	Changes Forecast	3
1	GENERAL	4
1.1	Scope	4
1.2	Introduction	4
2	PRELIMINARIES	4
3	SETTING UP FOR THE SECURITY PROOF	4
3.1	Theorem about <i>dominates</i>	4
3.2	Consistency Proof for <i>repState</i> and <i>absState</i>	5
3.3	Retrieving the Remaining Definitions of Constants	5
3.4	Components of Labelled Products	6
3.4.1	Type <i>Directory</i>	6
3.4.2	Type <i>TableSpec</i>	7
3.4.3	Type <i>Row</i>	8
3.4.4	Type <i>Data</i>	8
4	PROOF OF SECURITY OF <i>hide</i>	9
4.1	Proof Strategy	9
4.2	Proof of Security of <i>hideR</i>	10
4.2.1	<i>replaceData</i> Lemma	11
4.2.2	<i>cleanColCons</i> Lemma	12
4.2.3	<i>cleanRow</i> Lemma	13
4.2.4	<i>cleanTable</i> Lemma	16
4.2.5	<i>cleanDirectory</i> Lemma	22
4.2.6	<i>hideR</i> Lemma	24
4.3	Proof that the Invariant on the State is Maintained	26
4.4	<i>hide</i> Lemma	30
5	CLOSING DOWN	31
6	THE THEORY fef010	32
6.1	Parents	32
6.2	Children	32
6.3	Constants	32
6.4	Definitions	32
6.5	Theorems	32
7	INDEX	36

0.2 Document Cross References

- [1] DS/FMU/017. *Secure Database Technical Proposal*. High Assurance Team, ICL Secure Systems, WIN01, 21st January 1992.
- [2] DS/FMU/FEF/007. *Proof Strategy*. G.M. Prout, ICL Secure Systems, WIN01.

0.3 Changes History

Issue Revision : 2.5 (5 December 2009) Consistency proof for *dominates* extended because of inclusion of *glb*.

Issue 2.5 Removed dependency on ICL logo font

0.4 Changes Forecast

None.

1 GENERAL

1.1 Scope

This document provides a formal proof that the component *hide* satisfies its critical requirement, as specified in the proof strategy [2]. It constitutes part of deliverable D6 of work package 1c, as given in section 7 of the Secure Database Technical Proposal, [1].

1.2 Introduction

This document is a proof script which provides a formal proof of the first conjunct of *Lemma1*, the requirement on the critical components *hide* and *updateState*, described in the proof strategy document [2].

Lemma1

| $?\vdash \quad \textit{hide} \in \textit{secureHide} \wedge (\textit{hide}, \textit{updateState}) \in \textit{secureUpdate}$

In this document, we give a proof of

| $?\vdash \quad \textit{hide} \in \textit{secureHide}$

First, we define a property *secureHideR* on the representation type, and then prove

| $?\vdash \quad \textit{hideR} \in \textit{secureHideR}$

We then prove that *hideR* maintains the state invariant. From these two results, we prove that *hide* is secure.

2 PRELIMINARIES

The following ProofPower instructions set up the new theory *fef010*.

```
SML
| open_theory "fef009";
| (force_delete_theory "fef010" handle _ => ());
| new_theory "fef010";
| push_merge_pcs["hol","wrk049","'pair1"];

```

3 SETTING UP FOR THE SECURITY PROOF

3.1 Theorem about *dominates*

A theorem which just states the transitivity property of *dominates*.

SML

```
| val dominates_trans = prove_rule[dominates_def]
|    $\ulcorner \forall x y z \bullet x \text{ dominates } y \wedge y \text{ dominates } z \Rightarrow x \text{ dominates } z \urcorner$ ;
```

HOL output

```
| dominates_trans =  $\vdash \forall x y z \bullet x \text{ dominates } y \wedge y \text{ dominates } z \Rightarrow x \text{ dominates } z$ 
```

3.2 Consistency Proof for *repState* and *absState*

We prove the consistency of *repState* and *absState* and retrieve their definitions with the consistency obligation satisfied.

SML

```
| push_consistency_goal  $\ulcorner \text{repState} \urcorner$ ;
| a(strip_asm_tac (simple_ $\Rightarrow$ _match_mp_rule type_lemmas_thm state_type_def));
| a( $\exists$ _tac  $\ulcorner (\text{rep}, \text{abs}) \urcorner$  THEN asm_rewrite_tac[]);
| a  $\wedge$ _tac;
```

SML

```
| (* *** Goal "1" *** *)
| a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
| a(POP_ASM_T (ante_tac o app_fun_rule  $\ulcorner \text{abs} \urcorner$ ) THEN asm_rewrite_tac[]);
| (* *** Goal "2" *** *)
| a(REPEAT  $\forall$ _tac THEN  $\Rightarrow$ _tac);
| a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
| a(POP_ASM_T (ante_tac o app_fun_rule  $\ulcorner \text{rep} \urcorner$ ) THEN asm_rewrite_tac[]);
| save_consistency_thm  $\ulcorner \text{repState} \urcorner$  (pop_thm());
| val repState_absState_def = get_spec  $\ulcorner \text{repState} \urcorner$ ;
```

3.3 Retrieving the Remaining Definitions of Constants

SML

```
| val hide_def = get_spec  $\ulcorner \text{hide} \urcorner$ ;
| val hideR_def = get_spec  $\ulcorner \text{hideR} \urcorner$ ;
| val secureHide_def = get_spec  $\ulcorner \text{secureHide} \urcorner$ ;
| val cleanDirectory_def = get_spec  $\ulcorner \text{cleanDirectory} \urcorner$ ;
| val cleanTable_def = conv_rule(MAP_C let_conv)(get_spec  $\ulcorner \text{cleanTable} \urcorner$ );
| val cleanColCons_def = conv_rule(MAP_C let_conv)(get_spec  $\ulcorner \text{cleanColCons} \urcorner$ );
| val cleanRows_def = get_spec  $\ulcorner \text{cleanRows} \urcorner$ ;
| val cleanRow_def = get_spec  $\ulcorner \text{cleanRow} \urcorner$ ;
| val replaceData_def = get_spec  $\ulcorner \text{replaceData} \urcorner$ ;
| val filterRow_def = get_spec  $\ulcorner \text{filterRow} \urcorner$ ;
```

3.4 Components of Labelled Products

These theorems state that if two things of labelled product type are equal, then their components must be equal.

3.4.1 Type *Directory*

SML

```

push_goal([],Γ∀ dir1 dir2 •
  dir1 = dir2 ⇔
  Dir_tables dir1 = Dir_tables dir2
  ∧ Dir_exist dir1 = Dir_exist dir2
  ∧ Dir_class dir1 = Dir_class dir2¬);
a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
a(lemma_tacΓ(dir1 = MkDirectory (Dir_tables dir1) (Dir_exist dir1) (Dir_class dir1))
  ∧ (dir2 = MkDirectory (Dir_tables dir2) (Dir_exist dir2) (Dir_class dir2))¬);
(* *** Goal "1" *** *)
a(rewrite_tac[get_specΓMkDirectory¬]);
(* *** Goal "2" *** *)
a(POP_ASM_T once_rewrite_thm_tac);
a(POP_ASM_T once_rewrite_thm_tac);
a(asm_rewrite_tac[]);
val dir_components = save_pop_thm"dir_components";

```

HOL output

```

dir_components =
  ⊢ ∀ dir1 dir2
  • dir1 = dir2
  ⇔ Dir_tables dir1 = Dir_tables dir2
  ∧ Dir_exist dir1 = Dir_exist dir2
  ∧ Dir_class dir1 = Dir_class dir2

```

3.4.2 Type TableSpec

SML

```

push_goal([],Γ∀ t1 t2 •
  t1 = t2 ⇔
  TS_class t1 = TS_class t2
∧ TS_maxRow t1 = TS_maxRow t2
∧ TS_colspecs t1 = TS_colspecs t2
∧ TS_cons t1 = TS_cons t2
∧ TS_rows t1 = TS_rows t2¬);
a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
a(lemma_tacΓ(t1 = MkTableSpec(TS_class t1) (TS_maxRow t1) (TS_colspecs t1)
  (TS_cons t1)(TS_rows t1))
  ∧ (t2 = MkTableSpec(TS_class t2) (TS_maxRow t2) (TS_colspecs t2)
  (TS_cons t2)(TS_rows t2))¬);
(* *** Goal "1" *** *)
a(rewrite_tac[get_specΓMkTableSpec¬]);
(* *** Goal "2" *** *)
a(POP_ASM_T once_rewrite_thm_tac);
a(POP_ASM_T once_rewrite_thm_tac);
a(asm_rewrite_tac[]);
val tab_components = save_pop_thm"tab_components";

```

HOL output

```

tab_components =
| ⊢ ∀ t1 t2
| • t1 = t2
|   ⇔ TS_class t1 = TS_class t2
|     ∧ TS_maxRow t1 = TS_maxRow t2
|     ∧ TS_colspecs t1 = TS_colspecs t2
|     ∧ TS_cons t1 = TS_cons t2
|     ∧ TS_rows t1 = TS_rows t2

```

3.4.3 Type Row

SML

```

push_goal([],Γ∀ r1 r2 •
  r1 = r2 ⇔
  R_exist r1 = R_exist r2
  ∧ R_data r1 = R_data r2⊥);
a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
a(lemma_tacΓ(r1 = MkRow(R_exist r1) (R_data r1))
  ∧ (r2 = MkRow(R_exist r2) (R_data r2))⊥);
(* *** Goal "1" *** *)
a(rewrite_tac[get_specΓMkRow⊥]);
(* *** Goal "2" *** *)
a(POP_ASM_T once_rewrite_thm_tac);
a(POP_ASM_T once_rewrite_thm_tac);
a(asm_rewrite_tac[]);
val row_components = save_pop_thm"row_components";

```

HOL output

```

row_components =
  ⊢ ∀ r1 r2
  • r1 = r2 ⇔ R_exist r1 = R_exist r2 ∧ R_data r1 = R_data r2

```

3.4.4 Type Data

SML

```

push_goal([],Γ∀ d1 d2 •
  d1 = d2 ⇔
  Dat_class d1 = Dat_class d2
  ∧ Dat_item d1 = Dat_item d2⊥);
a(REPEAT strip_tac THEN_TRY asm_rewrite_tac[]);
a(lemma_tacΓ(d1 = MkData(Dat_class d1) (Dat_item d1))
  ∧ (d2 = MkData(Dat_class d2) (Dat_item d2))⊥);
(* *** Goal "1" *** *)
a(rewrite_tac[get_specΓMkData⊥]);
(* *** Goal "2" *** *)
a(POP_ASM_T once_rewrite_thm_tac);
a(POP_ASM_T once_rewrite_thm_tac);
a(asm_rewrite_tac[]);
val data_components = save_pop_thm"data_components";

```

HOL output

```

val data_components =
  ⊢ ∀ d1 d2
  • d1 = d2 ⇔ Dat_class d1 = Dat_class d2 ∧ Dat_item d1 = Dat_item d2

```

4 PROOF OF SECURITY OF *hide*

4.1 Proof Strategy

First we define *secureHideR* a property on things of type $Class \times StateR \rightarrow StateR$

HOL Constant

$\mathbf{secureHideR} : (Class \times StateR \rightarrow StateR) \mathbb{P}$ <hr style="border: 0.5px solid black; margin: 5px 0;"/> $\forall h : Class \times StateR \rightarrow StateR \bullet$ $h \in \mathit{secureHideR}$ \Leftrightarrow $\forall c_1 c_2 : Class; s_1 s_2 : StateR \bullet$ $h(c_1, s_1) = h(c_1, s_2)$ \wedge $c_1 \text{ dominates } c_2$ \Rightarrow $h(c_2, s_1) = h(c_2, s_2)$
--

We simplify some of the properties of the representation and abstraction functions.

SML

```
| val isState_lemma = all_∀_intro(nth 4 (strip_∧_rule(all_∀_elim repState_absState_def)));
```

HOL output

```
| isState_lemma = ⊢ ∀ s • isState (repState s)
```

SML

```
| val isState_lemma1 = all_∀_intro(nth 1 (strip_∧_rule(all_∀_elim repState_absState_def)));
```

HOL output

```
| isState_lemma1 = ⊢ isState r ⇒ repState (absState r) = r
```

SML

```
| val isState_lemma2 = all_∀_intro(nth 3 (strip_∧_rule(all_∀_elim repState_absState_def)));
```

HOL output

```
| isState_lemma2 =
| ⊢ ∀ r1 r2
| • isState r1 ∧ isState r2 ⇒ (absState r1 = absState r2 ⇔ r1 = r2)
```

Next, we prove that if *hideR* is a member of *secureHideR* and *hideR* maintains the invariant on the representation state, then *hide* is a member of *secureHide*.

SML

```

push_goal([],⌈(hideR ∈ secureHideR ∧ ∀ clear s • isState s ⇒ isState(hideR(clear,s)))
  ⇒ hide ∈ secureHide⌋);
a(rewrite_tac[secureHide_def,hide_def,get_spec⌈secureHideR⌋] THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 4 (ante_tac o list_∀_elim[⌈c1⌋,⌈c2⌋,⌈repState s1⌋,⌈repState s2⌋]));
a(asm_rewrite_tac[] THEN ⇒_T asm_tac);
a(GET_NTH_ASM_T 4 (asm_tac o rewrite_rule[isState_lemma] o
  list_∀_elim[⌈c1⌋,⌈repState s1⌋]));
a(GET_NTH_ASM_T 5 (asm_tac o rewrite_rule[isState_lemma] o
  list_∀_elim[⌈c1⌋,⌈repState s2⌋]));
a(strip_asm_tac (list_∀_elim[⌈hideR (c1, repState s1)⌋,⌈hideR (c1, repState s2)⌋]
  isState_lemma2));
a(DROP_NTH_ASM_T 4 ante_tac THEN asm_rewrite_tac[] THEN ⇒_tac);
a(GET_NTH_ASM_T 7 (asm_tac o rewrite_rule[isState_lemma] o
  list_∀_elim[⌈c2⌋,⌈repState s1⌋]));
a(GET_NTH_ASM_T 8 (asm_tac o rewrite_rule[isState_lemma] o
  list_∀_elim[⌈c2⌋,⌈repState s2⌋]));
a(strip_asm_tac (list_∀_elim[⌈hideR (c2, repState s1)⌋,⌈hideR (c2, repState s2)⌋]
  isState_lemma2));
val hideR_hide_lemma = save_pop_thm"hideR_hide_lemma";

```

HOL output

```

hideR_hide_lemma =
⊢ hideR ∈ secureHideR
  ∧ (∀ clear s • isState s ⇒ isState (hideR (clear, s)))
  ⇒ hide ∈ secureHide

```

4.2 Proof of Security of *hideR*

The proof is broken down into a series of lemmas.

4.2.1 *replaceData* Lemma

SML

```

push_goal([], $\Gamma \forall c_1 c_2 \bullet c_1 \text{ dominates } c_2$ 
   $\Rightarrow \forall d_1 d_2 \bullet$ 
    replaceData  $c_1 d_1 = \text{replaceData } c_1 d_2$ 
     $\Rightarrow$ 
    replaceData  $c_2 d_1 = \text{replaceData } c_2 d_2$ );
a(REPEAT strip_tac);
a(POP_ASM_T ante_tac THEN rewrite_tac[replaceData_def]);
a(cases_tac $\Gamma c_2 \text{ dominates } \text{Dat\_class } d_1 \neg$  THEN cases_tac $\Gamma c_2 \text{ dominates } \text{Dat\_class } d_2 \neg$ 
  THEN asm_rewrite_tac());
(* *** Goal "1" *** *)
a(lemma_tac $\Gamma (c_1 \text{ dominates } \text{Dat\_class } d_1) \wedge (c_1 \text{ dominates } \text{Dat\_class } d_2) \neg$ 
  THEN_LIST[basic_res_tac2 3[dominates_trans],asm_rewrite_tac]);

```

SML

```

(* *** Goal "2" *** *)
a(lemma_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_1 \neg$  THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,4][dominates_trans],asm_rewrite_tac]);
a(cases_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_2 \neg$  THEN asm_rewrite_tac());
a $\Rightarrow$ _tac;
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac());
(* *** Goal "3" *** *)
a(lemma_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_2 \neg$  THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,4][dominates_trans],asm_rewrite_tac]);
a(cases_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_1 \neg$  THEN asm_rewrite_tac());
a $\Rightarrow$ _tac;
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac());

```

SML

```

(* *** Goal "4" *** *)
a(cases_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_1 \neg$  THEN cases_tac $\Gamma c_1 \text{ dominates } \text{Dat\_class } d_2 \neg$ 
  THEN asm_rewrite_tac());
(* *** Goal "4.1" *** *)
a $\Rightarrow$ _tac;
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac());
(* *** Goal "4.2" *** *)
a( $\Rightarrow$ _T(strip_asm_tac o rewrite_rule[data_components,get_spec $\Gamma \text{MkData}$ ])); ;
a(asm_rewrite_tac());
(* *** Goal "4.3" *** *)
a( $\Rightarrow$ _T(strip_asm_tac o rewrite_rule[data_components,get_spec $\Gamma \text{MkData}$ ])); ;
a(asm_rewrite_tac());
val replaceData_lemma = save_pop_thm"replaceData_lemma";

```

HOL output

```

replaceData_lemma =
| ⊢ ∀ c1 c2
|   • c1 dominates c2
|     ⇒ (∀ d1 d2
|       • replaceData c1 d1 = replaceData c1 d2
|         ⇒ replaceData c2 d1 = replaceData c2 d2)

```

4.2.2 cleanColCons Lemma

SML

```

push_goal([], ⊢ ∀ c1 c2 t1 t2 •
|   (c1 dominates c2
|     ∧ cleanColCons c1 t1 = cleanColCons c1 t2)
|     ⇒ cleanColCons c2 t1 = cleanColCons c2 t2);
a(REPEAT strip_tac);
a(POP_ASM_T ante_tac THEN rewrite_tac[cleanColCons_def, ▷_def, dom_def]);
a(PC_T "hol1"(REPEAT strip_tac));
(* *** Goal "1" *** *)
a(lemma_tac ⊢ c1 dominates CC_exist (Snd x));
(* *** Goal "1.1" *** *)
a(ante_tac(list_∀_elim[⊢ c1, ⊢ c2, ⊢ CC_exist (Snd x)]dominates_def) THEN asm_rewrite_tac[]);
a(REPEAT strip_tac);
(* *** Goal "1.2" *** *)
a(PC_T"hol1"(DROP_NTH_ASM_T 5 (strip_asm_tac o ∀_elim ⊢ x)));

```

SML

```

(* *** Goal "2" *** *)
a(lemma_tac ⊢ c1 dominates CC_exist (Snd x));
(* *** Goal "2.1" *** *)
a(ante_tac(list_∀_elim[⊢ c1, ⊢ c2, ⊢ CC_exist (Snd x)]dominates_def) THEN asm_rewrite_tac[]);
a(REPEAT strip_tac);
(* *** Goal "2.2" *** *)
a(PC_T"hol1"(DROP_NTH_ASM_T 5 (strip_asm_tac o ∀_elim ⊢ x)));
(* *** Goal "3" *** *)
a(∃_tac ⊢ y THEN asm_rewrite_tac[]);
a(lemma_tac ⊢ c1 dominates CC_exist y);
(* *** Goal "3.1" *** *)
a(ante_tac(list_∀_elim[⊢ c1, ⊢ c2, ⊢ CC_exist y]dominates_def) THEN asm_rewrite_tac[]);
a(REPEAT strip_tac);
(* *** Goal "3.2" *** *)
a(PC_T"hol1"(DROP_NTH_ASM_T 5 (strip_asm_tac o ∀_elim ⊢ (CS_consGroup x, y))));

```

SML

```

(* *** Goal "4" *** *)
a( $\exists$ -tac $\ulcorner$ y $\urcorner$  THEN asm_rewrite_tac[]);
a(lemma_tac  $\ulcorner$ c1 dominates CC_exist y $\urcorner$ );
(* *** Goal "4.1" *** *)
a(ante_tac(list_∇_elim $\ulcorner$ c1 $\urcorner$ , $\ulcorner$ c2 $\urcorner$ , $\ulcorner$ CC_exist y $\urcorner$ )dominates_def) THEN asm_rewrite_tac[]);
a(REPEAT strip_tac);
(* *** Goal "4.2" *** *)
a(PC_T"hol1"(DROP_NTH_ASM_T 5 (strip_asm_tac o ∇_elim $\ulcorner$ (CS_consGroup x, y) $\urcorner$ )));
val cleanColCons_lemma = save_pop_thm"cleanColCons_lemma";

```

HOL output

```

cleanColCons_lemma =
|  $\vdash \forall c_1 c_2 t_1 t_2$ 
| •  $c_1$  dominates  $c_2 \wedge$  cleanColCons  $c_1 t_1 =$  cleanColCons  $c_1 t_2$ 
|    $\Rightarrow$  cleanColCons  $c_2 t_1 =$  cleanColCons  $c_2 t_2$ 

```

4.2.3 cleanRow Lemma

First, two results about *cleanColCons*.

SML

```

push_goal([], $\ulcorner$  $\forall c_1 c_2 t \bullet c_1$  dominates  $c_2$ 
 $\Rightarrow$  (Fst(cleanColCons  $c_2 t) \subseteq$  Fst(cleanColCons  $c_1 t)$ 
 $\wedge$  Snd(cleanColCons  $c_2 t) \subseteq$  Snd(cleanColCons  $c_1 t)) $\urcorner$ );
a(rewrite_tac[rel_ext_clauses,sets_ext_clauses,cleanColCons_def, $\triangleright$ -thm,dom_def]
THEN REPEAT strip_tac);
(* *** Goal "1" *** *)
a(contr_tac THEN basic_res_tac4 2 [1][3,4][][][dominates_trans]);
(* *** Goal "2" *** *)
a( $\exists$ -tac $\ulcorner$ y $\urcorner$  THEN asm_rewrite_tac[]);
a(contr_tac THEN basic_res_tac4 2 [1][3,4][][][dominates_trans]);
val  $\subseteq$ -cleanColCons_lemma = save_pop_thm" $\subseteq$ -cleanColCons_lemma";$ 
```

HOL output

```

 $\subseteq$ -cleanColCons_lemma =
|  $\vdash \forall c_1 c_2 t$ 
| •  $c_1$  dominates  $c_2$ 
|    $\Rightarrow$  Fst (cleanColCons  $c_2 t) \subseteq$  Fst (cleanColCons  $c_1 t)$ 
|    $\wedge$  Snd (cleanColCons  $c_2 t) \subseteq$  Snd (cleanColCons  $c_1 t)$ 

```

SML

```

push_goal([],Γ∀ c1 c2 t col • c1 dominates c2
  ⇒ (col ∈ Snd(cleanColCons c2 t) ⇒ col ∈ Snd(cleanColCons c1 t))∇);
a(REPEAT strip_tac);
a(basic_res_tac 2 [rewrite_rule[sets_ext_clauses] ⊆_cleanColCons_lemma]);
val ⊆_cleanColCons_lemma1 = save_pop_thm"⊆_cleanColCons_lemma1";

```

HOL output

```

⊆_cleanColCons_lemma1 =
| ⊢ ∀ c1 c2 t col
|   • c1 dominates c2
|     ⇒ col ∈ Snd (cleanColCons c2 t)
|     ⇒ col ∈ Snd (cleanColCons c1 t)

```

The main result about *CleanRow*.

SML

```

push_goal([],Γ∀ c1 c2 • c1 dominates c2
  ⇒ ∀ t1 t2 r1 r2 •
    (cleanColCons c1 t1 = cleanColCons c1 t2
      ∧ cleanColCons c2 t1 = cleanColCons c2 t2
      ∧ cleanRow c1 (Snd (cleanColCons c1 t1)) r1
        = cleanRow c1 (Snd (cleanColCons c1 t2)) r2)
    ⇒
      cleanRow c2 (Snd (cleanColCons c2 t1)) r1
        = cleanRow c2 (Snd (cleanColCons c2 t2)) r2∇);
a(REPEAT ∀_tac THEN strip_tac);
a(strip_asm_tac (list_∀_elim[Γc1∇,Γc2∇]replaceData_lemma));
a(rewrite_tac[cleanRow_def,row_components,get_specΓMkRow∇,filterRow_def,
  rel_ext_clauses,◁_thm,r_?_r_thm,graph_thm] THEN REPEAT strip_tac);
(* *** Goal "1" *** *)
a(strip_asm_tac (list_∀_elim[Γc1∇,Γc2∇,Γt1∇,Γc∇]⊆_cleanColCons_lemma1));
a(DROP_NTH_ASM_T 6 (asm_tac o list_∀_elim[Γx∇,ΓreplaceData c1 z∇]));
a(LEMMA_TΓ∃ z'
  • ((∃ c • c ∈ Snd (cleanColCons c1 t1) ∧ CS_posn c = x)
    ∧ (x, z') ∈ R_data r1)
    ∧ replaceData c1 z = replaceData c1 z'∇asm_tac);
(* *** Goal "1.1" *** *)
a(∃_tacΓz∇ THEN asm_prove_tac[]);

```

SML

```

(* *** Goal "1.2" *** *)
a(DROP_NTH_ASM_T 2 ante_tac THEN POP_ASM_T rewrite_thm_tac);
a  $\Rightarrow$  _tac;
a(DROP_NTH_ASM_T 13(strip_asm_tac o list_ $\forall$ _elim[ $\Gamma$  z $\neg$ , $\Gamma$  z' $\neg$ ]));
a( $\exists$ _tac $\Gamma$  z' $\neg$  THEN asm_rewrite_tac[]);
a( $\exists$ _tac $\Gamma$  c $\neg$  THEN asm_rewrite_tac[]);
a(DROP_NTH_ASM_T 10 ante_tac THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "2" *** *)
a(strip_asm_tac (list_ $\forall$ _elim[ $\Gamma$  c $_1$  $\neg$ , $\Gamma$  c $_2$  $\neg$ , $\Gamma$  t $_2$  $\neg$ , $\Gamma$  c $\neg$ ]  $\sqsubseteq$  cleanColCons_lemma1));
a(DROP_NTH_ASM_T 6 (asm_tac o list_ $\forall$ _elim[ $\Gamma$  x $\neg$ , $\Gamma$  replaceData c $_1$  z' $\neg$ ]));
a(LEMMA_T $\Gamma$   $\exists$  z'
  • (( $\exists$  c • c  $\in$  Snd (cleanColCons c $_1$  t $_2$ )  $\wedge$  CS_posn c = x)
     $\wedge$  (x, z')  $\in$  R_data r $_2$ )
     $\wedge$  replaceData c $_1$  z = replaceData c $_1$  z' $\neg$ asm_tac);
(* *** Goal "2.1" *** *)
a( $\exists$ _tac $\Gamma$  z $\neg$  THEN asm_prove_tac[]);

```

SML

```

(* *** Goal "2.2" *** *)
a(DROP_NTH_ASM_T 2 ante_tac THEN POP_ASM_T rewrite_thm_tac);
a  $\Rightarrow$  _tac;
a(DROP_NTH_ASM_T 13(strip_asm_tac o list_ $\forall$ _elim[ $\Gamma$  z $\neg$ , $\Gamma$  z' $\neg$ ]));
a( $\exists$ _tac $\Gamma$  z' $\neg$  THEN asm_rewrite_tac[]);
a( $\exists$ _tac $\Gamma$  c $\neg$  THEN asm_rewrite_tac[]);
val cleanRow_lemma = save_pop_thm"cleanRow_lemma";

```

HOL output

```

cleanRow_lemma =
|  $\vdash \forall c_1 c_2$ 
| • c $_1$  dominates c $_2$ 
|  $\Rightarrow (\forall t_1 t_2 r_1 r_2$ 
| • cleanColCons c $_1$  t $_1$  = cleanColCons c $_1$  t $_2$ 
|  $\wedge$  cleanColCons c $_2$  t $_1$  = cleanColCons c $_2$  t $_2$ 
|  $\wedge$  cleanRow c $_1$  (Snd (cleanColCons c $_1$  t $_1$ )) r $_1$ 
| = cleanRow c $_1$  (Snd (cleanColCons c $_1$  t $_2$ )) r $_2$ 
|  $\Rightarrow$  cleanRow c $_2$  (Snd (cleanColCons c $_2$  t $_1$ )) r $_1$ 
| = cleanRow c $_2$  (Snd (cleanColCons c $_2$  t $_2$ )) r $_2$ )

```

4.2.4 cleanTable Lemma

SML

```

| push_goal([],Γ∀ c1 c2 • c1 dominates c2 ⇒ ∀ t1 t2 •
|   cleanTable c1 t1 = cleanTable c1 t2 ⇒ cleanTable c2 t1 = cleanTable c2 t2∇);
| a(rewrite_tac[cleanTable_def] THEN REPEAT ∀_tac THEN strip_tac THEN REPEAT ∀_tac);
| a(cases_tacΓc2 dominates TS_class t1∇ THEN cases_tacΓc2 dominates TS_class t2∇
|   THEN asm_rewrite_tac[]);

```

SML

```

| (* *** Goal "1" *** *)
| a(lemma_tacΓc1 dominates TS_class t1∇ THEN_LIST
|   [contr_tac THEN basic_res_tac4 2 [1][3,4][dominates_trans],asm_rewrite_tac[]]);
| a(lemma_tacΓc1 dominates TS_class t2∇ THEN_LIST
|   [contr_tac THEN basic_res_tac4 2 [1][3,5][dominates_trans],asm_rewrite_tac[]]);
| a(rewrite_tac[tab_components,get_specΓMkTableSpec∇]);
| a(strip_tac THEN asm_rewrite_tac[]);
| a(strip_asm_tac (list_∇_elimΓcleanColCons c1 t1∇,ΓcleanColCons c1 t2∇pair_eq_thm1));
| a(rewrite_tac[taut_ruleΓ∀ a b c •(a ∧ (b ∧ c)) = ((b ∧ a) ∧ c)∇,pair_eq_thm1]);
| a(DROP_NTH_ASM_T 4 (fn _ => id_tac)
|   THEN DROP_NTH_ASM_T 3 (fn _ => id_tac));
| a(strip_asm_tac(list_∇_elimΓc1∇,Γc2∇,Γt1∇,Γt2∇cleanColCons_lemma));
| a(∧_tac THEN_LIST[asm_rewrite_tac[],id_tac]);

```

SML

```

| a(DROP_NTH_ASM_T 3 ante_tac);
| a(lemma_tacΓ∃ l • TS_rows t1 = l∇THEN_LIST
|   [prove_∃_tac,POP_ASM_T rewrite_thm_tac]);
| a(lemma_tacΓ∃ l1 • TS_rows t2 = l1∇THEN_LIST
|   [prove_∃_tac,POP_ASM_T rewrite_thm_tac]);
| a(intro_∇_tac(Γl1∇,Γl':Row LIST∇));
| a(rewrite_tac[cleanRows_def]);
| a(LIST_INDUCTION_TΓl∇asm_tac);
| (* *** Goal "1.1" *** *)
| a(rewrite_tac[map_def,map_null_thm]);
| a ∀_tac;
| a(LIST_INDUCTION_TΓl'∇asm_tac);

```

SML

```

(* *** Goal "1.1.1" *** *)
a(rewrite_tac[|-def]);
(* *** Goal "1.1.2" *** *)
a(rewrite_tac[|-def]);
a strip_tac;
a(cases_tacΓc2 dominates R_exist x∇ THEN asm_rewrite_tac[]);
(* *** Goal "1.1.2.1" *** *)
a(lemma_tacΓc1 dominates R_exist x∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,11]] [dominates_trans],asm_rewrite_tac[]);
(* *** Goal "1.1.2.2" *** *)
a(cases_tacΓc1 dominates R_exist x∇ THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "1.2" *** *)
a(rewrite_tac[∀_reorder_convΓ∀ l' x
  • Map
    (cleanRow c1 (Snd (cleanColCons c1 t1)))
    (Cons x l | {r|c1 dominates R_exist r})
  = Map
    (cleanRow c1 (Snd (cleanColCons c1 t2)))
    (l' | {r|c1 dominates R_exist r})
  ⇒ Map
    (cleanRow c2 (Snd (cleanColCons c2 t1)))
    (Cons x l | {r|c2 dominates R_exist r})
  = Map
    (cleanRow c2 (Snd (cleanColCons c2 t2)))
    (l' | {r|c2 dominates R_exist r})∇∀ x l'
  • Map
    (cleanRow c1 (Snd (cleanColCons c1 t1)))
    (Cons x l | {r|c1 dominates R_exist r})
  = Map
    (cleanRow c1 (Snd (cleanColCons c1 t2)))
    (l' | {r|c1 dominates R_exist r})
  ⇒ Map
    (cleanRow c2 (Snd (cleanColCons c2 t1)))
    (Cons x l | {r|c2 dominates R_exist r})
  = Map
    (cleanRow c2 (Snd (cleanColCons c2 t2)))
    (l' | {r|c2 dominates R_exist r})∇]);

```

SML

```

a  $\forall$ _tac;
a(LIST_INDUCTION_TΓ l'∇ asm_tac);
(* *** Goal "1.2.1" *** *)
a(rewrite_tac[ $\lambda$ _def] THEN  $\forall$ _tac);
a(cases_tacΓ c2 dominates R_exist x∇ THEN TOP_ASM_T rewrite_thm_tac);
(* *** Goal "1.2.1.1" *** *)
a(lemma_tacΓ c1 dominates R_exist x∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,4]] [dominates_trans],
  TOP_ASM_T rewrite_thm_tac);
a(rewrite_tac[map_def]);
(* *** Goal "1.2.1.2" *** *)
a(cases_tacΓ c1 dominates R_exist x∇ THEN TOP_ASM_T rewrite_thm_tac);
(* *** Goal "1.2.1.2.1" *** *)
a(rewrite_tac[map_def]);
(* *** Goal "1.2.1.2.2" *** *)
a(DROP_NTH_ASM_T  $\exists$ (ante_tac o  $\forall$ _elimΓ []:Row LIST∇));
a(rewrite_tac[map_def]);

```

SML

```

(* *** Goal "1.2.2" *** *)
a(rewrite_tac[ $\lambda$ _def]);
a(REPEAT  $\forall$ _tac);
a(cases_tacΓ c2 dominates R_exist x∇
  THEN TOP_ASM_T rewrite_thm_tac THEN cases_tacΓ c2 dominates R_exist x'∇
  THEN TOP_ASM_T rewrite_thm_tac);
(* *** Goal "1.2.2.1" *** *)
a(lemma_tacΓ c1 dominates R_exist x∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,14]] [dominates_trans],
  TOP_ASM_T rewrite_thm_tac);
a(lemma_tacΓ c1 dominates R_exist x'∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,14]] [dominates_trans],
  TOP_ASM_T rewrite_thm_tac);
a(rewrite_tac[map_def]);

```

SML

```

a(DROP_NTH_ASM_T 6 (asm_tac o  $\forall$ _elim $\ulcorner$ l' $\urcorner$ ));
a(strip_asm_tac(list_ $\forall$ _elim $\ulcorner$ c1 $\urcorner$ , $\ulcorner$ c2 $\urcorner$ )cleanRow_lemma));
a(POP_ASM_T (ante_tac o list_ $\forall$ _elim $\ulcorner$ t1 $\urcorner$ , $\ulcorner$ t2 $\urcorner$ , $\ulcorner$ x' $\urcorner$ , $\ulcorner$ x $\urcorner$ ));
a(asm_rewrite_tac[] THEN  $\Rightarrow$ _T asm_tac);
a strip_tac;
a(DROP_NTH_ASM_T 4 ante_tac THEN DROP_NTH_ASM_T 3 ante_tac
  THEN asm_rewrite_tac[] THEN REPEAT strip_tac);
(* *** Goal "1.2.2.2" *** *)
a(lemma_tac $\ulcorner$ c1 dominates R_exist x' $\urcorner$  THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,14]][[dominates_trans],
  TOP_ASM_T rewrite_thm_tac]);
a(cases_tac  $\ulcorner$ c1 dominates R_exist x' $\urcorner$  THEN asm_rewrite_tac[]);
(* *** Goal "1.2.2.2.1" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 2
  (ante_tac o rewrite_rule[cleanRow_def,row_components,get_spec $\ulcorner$ MkRow $\urcorner$ ]));
a strip_tac;
a(DROP_NTH_ASM_T 6 ante_tac THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "1.2.2.2.2" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 7(ante_tac o  $\forall$ _elim $\ulcorner$ Cons x l' $\urcorner$ ));
a(asm_rewrite_tac[map_def, $\ulcorner$ _def]);
(* *** Goal "1.2.2.3" *** *)
a(lemma_tac $\ulcorner$ c1 dominates R_exist x' $\urcorner$  THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,14]][[dominates_trans],
  TOP_ASM_T rewrite_thm_tac]);
a(cases_tac  $\ulcorner$ c1 dominates R_exist x' $\urcorner$  THEN asm_rewrite_tac[]);
(* *** Goal "1.2.2.3.1" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 2(ante_tac o
  rewrite_rule[cleanRow_def,row_components,get_spec $\ulcorner$ MkRow $\urcorner$ ]));
a strip_tac;
a(DROP_NTH_ASM_T 6 ante_tac THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "1.2.2.3.2" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 6(ante_tac o  $\forall$ _elim $\lceil x' \rceil$ ));
a(asm_rewrite_tac[map_def,  $\lceil$ _def]);
(* *** Goal "1.2.2.4" *** *)
a(cases_tac  $\lceil c_1$  dominates R_exist  $x' \rceil$  THEN cases_tac  $\lceil c_1$  dominates R_exist  $x' \rceil$ 
  THEN asm_rewrite_tac[]);
(* *** Goal "1.2.2.4.1" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 8(ante_tac o  $\forall$ _elim $\lceil l' \rceil$ ) THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "1.2.2.4.2" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 7(ante_tac o  $\forall$ _elim $\lceil$ Cons  $x l' \rceil$ ));
a(asm_rewrite_tac[map_def,  $\lceil$ _def]);
(* *** Goal "1.2.2.4.3" *** *)
a(rewrite_tac[map_def] THEN strip_tac);
a(DROP_NTH_ASM_T 6(ante_tac o  $\forall$ _elim $\lceil x' \rceil$ ));
a(asm_rewrite_tac[map_def,  $\lceil$ _def]);
(* *** Goal "1.2.2.4.4" *** *)
a(strip_tac THEN DROP_NTH_ASM_T 7(ante_tac o  $\forall$ _elim $\lceil l' \rceil$ ) THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "2" *** *)
a(lemma_tac $\lceil c_1$  dominates TS_class  $t_1 \rceil$  THEN_LIST
  [ $\lceil$ contr_tac THEN basic_res_tac4 2 [1][3,4][ $\lceil$ dominates_trans $\rceil$ ,asm_rewrite_tac $\rceil$ ]);
a(cases_tac $\lceil c_1$  dominates TS_class  $t_2 \rceil$  THEN asm_rewrite_tac[]);
(* *** Goal "2.1" *** *)
a(rewrite_tac[tab_components,get_spec $\lceil$ MkTableSpec $\rceil$ ]);
a strip_tac;
a(DROP_NTH_ASM_T 9 ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "2.2" *** *)
a(rewrite_tac[tab_components,get_spec $\lceil$ MkTableSpec $\rceil$ ]);
a(strip_tac THEN asm_rewrite_tac[]);
a(rewrite_tac[taut_rule $\lceil \forall a b c \bullet (a \wedge (b \wedge c)) = ((b \wedge a) \wedge c) \rceil$ ]);
a strip_tac;

```

SML

```

(* *** Goal "2.2.1" *** *)
a(ante_tac(list_∇_elim[ $\Gamma c_1 \neg$ ,  $\Gamma c_2 \neg$ ,  $\Gamma t_1 \neg$ ]  $\subseteq$  cleanColCons_lemma));
a(asm_rewrite_tac[ $\subseteq$ -∅_thm] THEN REPEAT strip_tac);
(* *** Goal "2.2.2" *** *)
a(POP_ASM_T ante_tac THEN
  rewrite_tac[cleanRows_def, map_null_thm,  $\lceil$ _thm $\lrcorner$ a, sets_ext_clauses]);
a(REPEAT strip_tac);

```

SML

```

(* *** Goal "2.2.2.1" *** *)
a(lemma_tac $\Gamma c_1$  dominates R_exist  $x \neg$  THEN_LIST
  [contr_tac THEN basic_res_tac $\lrcorner$  2 [1][3,13]] [[dominates_trans], id_tac]);
a(DROP_NTH_ASM_T 4 (strip_asm_tac o  $\forall$ _elim $\Gamma x \neg$ ));
(* *** Goal "2.2.2.2" *** *)
a(lemma_tac $\Gamma c_1$  dominates R_exist  $x' \neg$  THEN_LIST
  [contr_tac THEN basic_res_tac $\lrcorner$  2 [1][2,13]] [[dominates_trans], basic_res_tac1 1 []]);

```

SML

```

(* *** Goal "3" *** *)
a(lemma_tac $\Gamma c_1$  dominates TS_class  $t_2 \neg$  THEN_LIST
  [contr_tac THEN basic_res_tac $\lrcorner$  2 [1][2,4]] [[dominates_trans], asm_rewrite_tac []]);
a(cases_tac $\Gamma c_1$  dominates TS_class  $t_1 \neg$  THEN asm_rewrite_tac []);
(* *** Goal "3.1" *** *)
a(rewrite_tac[tab_components, get_spec $\Gamma$  MkTableSpec $\neg$ ]);
a strip_tac;
a(DROP_NTH_ASM_T 9 ante_tac THEN asm_rewrite_tac []);

```

SML

```

(* *** Goal "3.2" *** *)
a(rewrite_tac[tab_components, get_spec $\Gamma$  MkTableSpec $\neg$ ]);
a strip_tac;
a(DROP_NTH_ASM_T 5(rewrite_thm_tac o eq_sym_rule));
a(DROP_NTH_ASM_T 4(rewrite_thm_tac o eq_sym_rule));
a(rewrite_tac[taut_rule $\forall a b c \bullet (a \wedge (b \wedge c)) = ((b \wedge a) \wedge c) \neg$ ] THEN strip_tac);
(* *** Goal "3.2.1" *** *)
a(ante_tac(list_∇_elim[ $\Gamma c_1 \neg$ ,  $\Gamma c_2 \neg$ ,  $\Gamma t_2 \neg$ ]  $\subseteq$  cleanColCons_lemma));
a(DROP_NTH_ASM_T 3 (rewrite_thm_tac o eq_sym_rule));
a(DROP_NTH_ASM_T 2 (rewrite_thm_tac o eq_sym_rule));
a(asm_rewrite_tac[ $\subseteq$ -∅_thm1] THEN REPEAT strip_tac);

```

SML

```

(* *** Goal "3.2.2" *** *)
a(POP_ASM_T ante_tac THEN
  rewrite_tac[cleanRows_def, map_null_thm, |-thm4a, sets_ext_clauses]);
a(REPEAT strip_tac);
(* *** Goal "3.2.2.1" *** *)
a(lemma_tac $\Gamma$ c1 dominates R_exist x∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,11][[dominates_trans], id_tac]);
a(DROP_NTH_ASM_T 4 (strip_asm_tac o  $\forall$ _elim $\Gamma$ x∇));

```

SML

```

(* *** Goal "3.2.2.2" *** *)
a(lemma_tac $\Gamma$ c1 dominates R_exist x'∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,11][[dominates_trans], basic_res_tac1 1 []]);
(* *** Goal "4" *** *)
a(cases_tac $\Gamma$ c1 dominates TS_class t1∇ THEN cases_tac $\Gamma$ c1 dominates TS_class t2∇
  THEN asm_rewrite_tac[tab_components, get_spec $\Gamma$ MkTableSpec∇]
  THEN strip_tac THEN asm_rewrite_tac[]);
val cleanTable_lemma = save_pop_thm "cleanTable_lemma";

```

HOL output

```

cleanTable_lemma =
|  $\vdash \forall c_1 c_2$ 
| •  $c_1$  dominates  $c_2$ 
|  $\Rightarrow (\forall t_1 t_2$ 
| •  $cleanTable\ c_1\ t_1 = cleanTable\ c_1\ t_2$ 
|  $\Rightarrow cleanTable\ c_2\ t_1 = cleanTable\ c_2\ t_2)$ 

```

4.2.5 cleanDirectory Lemma

SML

```

push_goal([],  $\Gamma \forall c_1 c_2 \bullet c_1$  dominates  $c_2 \Rightarrow \forall d_1 d_2 \bullet$ 
  cleanDirectory  $c_1\ d_1 = cleanDirectory\ c_1\ d_2$ 
 $\Rightarrow cleanDirectory\ c_2\ d_1 = cleanDirectory\ c_2\ d_2$ ∇);
a(REPEAT strip_tac);
a(POP_ASM_T ante_tac THEN rewrite_tac[cleanDirectory_def]);
a(cases_tac $\Gamma$ c2 dominates Dir_class d1∇ THEN cases_tac $\Gamma$ c2 dominates Dir_class d2∇
  THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "1" *** *)
a(lemma_tacΓc1 dominates Dir_class d1∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,4][][dominates_trans],asm_rewrite_tac[]]);
a(lemma_tacΓc1 dominates Dir_class d2∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,5][][dominates_trans],asm_rewrite_tac[]]);
a(rewrite_tac[dir_components,get_specΓMkDirectory∇] THEN REPEAT strip_tac);
a(strip_asm_tac(list_∇_elimΓc1∇,Γc2∇]cleanTable_lemma));
a(ante_tac (list_∇_elimΓcleanTable c1∇,ΓcleanTable c2∇,
  ΓDir_tables d1∇,ΓDir_tables d2∇]rel_thm) THEN asm_rewrite_tac[]);

```

SML

```

(* *** Goal "2" *** *)
a(lemma_tacΓc1 dominates Dir_class d1∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][3,4][][dominates_trans],asm_rewrite_tac[]]);
a(cases_tacΓc1 dominates Dir_class d2∇ THEN asm_rewrite_tac[]);
(* *** Goal "2.1" *** *)
a(rewrite_tac[dir_components,get_specΓMkDirectory∇] THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 7 ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "2.2" *** *)
a(rewrite_tac[dir_components,get_specΓMkDirectory∇] THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 3 ante_tac THEN rewrite_tac[graph_thm,r_∅_r_thm,rel_ext_clauses]);
a(REPEAT strip_tac);
a(DROP_NTH_ASM_T 2 (strip_asm_tac o list_∇_elimΓx∇,ΓcleanTable c1 z∇));
a(POP_ASM_T (strip_asm_tac o ∇_elimΓz∇));

```

SML

```

(* *** Goal "3" *** *)
a(lemma_tacΓc1 dominates Dir_class d2∇ THEN_LIST
  [contr_tac THEN basic_res_tac4 2 [1][2,4][][dominates_trans],asm_rewrite_tac[]]);
a(cases_tacΓc1 dominates Dir_class d1∇ THEN asm_rewrite_tac[]);
(* *** Goal "3.1" *** *)
a(rewrite_tac[dir_components,get_specΓMkDirectory∇] THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 7 ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "3.2" *** *)
a(rewrite_tac[dir_components,get_specΓMkDirectory∇] THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 3 ante_tac THEN rewrite_tac[graph_thm,r_∅_r_thm,rel_ext_clauses]);

```

SML

```

| a(REPEAT strip_tac);
| a(DROP_NTH_ASM_T 2 (strip_asm_tac o list_∀_elim[⌈x⌋,⌈cleanTable c1 z⌋]));
| a(POP_ASM_T (strip_asm_tac o ∀_elim⌈z⌋));
| (* *** Goal "4" *** *)
| a(cases_tac⌈c1 dominates Dir_class d1⌋ THEN cases_tac⌈c1 dominates Dir_class d2⌋
|   THEN asm_rewrite_tac[dir_components,get_spec⌈MkDirectory⌋]
|   THEN REPEAT strip_tac);
| val cleanDirectory_lemma = save_pop_thm"cleanDirectory_lemma";

```

HOL output

```

| cleanDirectory_lemma =
| ⊢ ∀ c1 c2
|   • c1 dominates c2
|     ⇒ (∀ d1 d2
|       • cleanDirectory c1 d1 = cleanDirectory c1 d2
|         ⇒ cleanDirectory c2 d1 = cleanDirectory c2 d2)

```

4.2.6 hideR Lemma

First, two lemmas about directories.

SML

```

| push_goal([],⌈∀ c1 c2 s x y • c1 dominates c2 ⇒
|   ((x,y) ∈ s ▷ {dir|c2 dominates Dir_exist dir}
|   ⇒ (x,y) ∈ s ▷ {dir|c1 dominates Dir_exist dir})⌋);
| a(rewrite_tac[▷_thm] THEN REPEAT strip_tac);
| a(contr_tac THEN basic_res_tac4 2 [1][3,4][[]][dominates_trans]);
| val ⊆_dir_lemma = save_pop_thm"⊆_dir_lemma";

```

HOL output

```

| ⊆_dir_lemma =
| ⊢ ∀ c1 c2 s x y
|   • c1 dominates c2
|     ⇒ (x, y) ∈ s ▷ {dir|c2 dominates Dir_exist dir}
|     ⇒ (x, y) ∈ s ▷ {dir|c1 dominates Dir_exist dir}

```

SML

```

push_goal([],Γ∀ c c1 s s1 x z z1
  • (cleanDirectory c1 z = cleanDirectory c1 z1
    ∧ (x, z) ∈ s ▷ {dir|c1 dominates Dir_exist dir}
    ∧ (x, z1) ∈ s1 ▷ {dir|c dominates Dir_exist dir})
    ⇒ (x, z1) ∈ s1 ▷ {dir|c1 dominates Dir_exist dir}¬);
a(rewrite_tac[cleanDirectory_def,dir_components,get_specΓMkDirectory¬,▷_thm]
  THEN REPEAT strip_tac);
a(DROP_NTH_ASM_T 4 ante_tac THEN asm_rewrite_tac[]);
val dir_class_preserved_lemma = save_pop_thm"dir_class_preserved_lemma";

```

HOL output

```

dir_class_preserved_lemma =
| ⊢ ∀ c c1 s s1 x z z1
  • cleanDirectory c1 z = cleanDirectory c1 z1
    ∧ (x, z) ∈ s ▷ {dir|c1 dominates Dir_exist dir}
    ∧ (x, z1) ∈ s1 ▷ {dir|c dominates Dir_exist dir}
    ⇒ (x, z1) ∈ s1 ▷ {dir|c1 dominates Dir_exist dir}

```

Now the main result of this section that *hideR* is a member of *secureHideR*.

SML

```

push_goal([],ΓhideR ∈ secureHideR¬);
a(rewrite_tac[get_specΓhideR¬,get_specΓsecureHideR¬]);
a(REPEAT strip_tac);
a(strip_asm_tac(list_∀_elim[Γc1¬,Γc2¬]cleanDirectory_lemma));
a(DROP_NTH_ASM_T 3 ante_tac);
a(rewrite_tac[r_9_r_thm,rel_ext_clauses,graph_thm]THEN REPEAT strip_tac);
(* *** Goal "1" *** *)
a(strip_asm_tac(list_∀_elim[Γc1¬,Γc2¬,Γs1¬,Γx¬,Γz¬]⊆_dir_lemma));
a(DROP_NTH_ASM_T 4 (asm_tac o list_∀_elim[Γx¬,ΓcleanDirectory c1 z¬]));
a(LEMMA_TΓ∃ z'
  • (x, z') ∈ s1 ▷ {dir|c1 dominates Dir_exist dir}
    ∧ cleanDirectory c1 z = cleanDirectory c1 z'¬asm_tac);
(* *** Goal "1.1" *** *)
a(∃_tacΓz¬ THEN asm_prove_tac[]);

```

SML

```

(* *** Goal "1.2" *** *)
a(DROP_NTH_ASM_T 2 ante_tac THEN POP_ASM_T rewrite_thm_tac);
a  $\Rightarrow$  _tac;
a(DROP_NTH_ASM_T 6(strip_asm_tac o list_ $\forall$ _elim[ $\Gamma$  z $\neg$ ,  $\Gamma$  z' $\neg$ ]));
a( $\exists$ _tac $\Gamma$  z' $\neg$  THEN asm_rewrite_tac[]);
a(strip_asm_tac (list_ $\forall$ _elim[ $\Gamma$  c1 $\neg$ ,  $\Gamma$  c2 $\neg$ ,  $\Gamma$  s1 $\neg$ ,  $\Gamma$  s2 $\neg$ ,  $\Gamma$  x $\neg$ ,  $\Gamma$  z $\neg$ ,  $\Gamma$  z' $\neg$ ]
  dir_class_preserved_lemma));
(* *** Goal "2" *** *)
a(strip_asm_tac (list_ $\forall$ _elim[ $\Gamma$  c1 $\neg$ ,  $\Gamma$  c2 $\neg$ ,  $\Gamma$  s2 $\neg$ ,  $\Gamma$  x $\neg$ ,  $\Gamma$  z $\neg$ ]  $\subseteq$  _dir_lemma));
a(DROP_NTH_ASM_T 4 (asm_tac o list_ $\forall$ _elim[ $\Gamma$  x $\neg$ ,  $\Gamma$  cleanDirectory c1 z $\neg$ ]));
a(LEMMA_T $\Gamma$   $\exists$  z'
  • (x, z')  $\in$  s2  $\triangleright$  {dir | c1 dominates Dir_exist dir}
     $\wedge$  cleanDirectory c1 z = cleanDirectory c1 z' $\neg$ asm_tac);
(* *** Goal "2.1" *** *)
a( $\exists$ _tac $\Gamma$  z $\neg$  THEN asm_prove_tac[]);

```

SML

```

(* *** Goal "2.2" *** *)
a(DROP_NTH_ASM_T 2 ante_tac THEN POP_ASM_T rewrite_thm_tac);
a  $\Rightarrow$  _tac;
a(DROP_NTH_ASM_T 6(strip_asm_tac o list_ $\forall$ _elim[ $\Gamma$  z $\neg$ ,  $\Gamma$  z' $\neg$ ]));
a( $\exists$ _tac $\Gamma$  z' $\neg$  THEN asm_rewrite_tac[]);
a(strip_asm_tac (list_ $\forall$ _elim[ $\Gamma$  c1 $\neg$ ,  $\Gamma$  c2 $\neg$ ,  $\Gamma$  s2 $\neg$ ,  $\Gamma$  s1 $\neg$ ,  $\Gamma$  x $\neg$ ,  $\Gamma$  z $\neg$ ,  $\Gamma$  z' $\neg$ ]
  dir_class_preserved_lemma));
val secureHideR_lemma = save_pop_thm"secureHideR_lemma";

```

HOL output

```

secureHideR_lemma =  $\vdash$  hideR  $\in$  secureHideR

```

4.3 Proof that the Invariant on the State is Maintained

We need a series of auxiliary results of the form *replaceData*, etc., is partial.

SML

```

push_goal([],  $\Gamma$   $\forall$  clear • Graph (cleanDirectory clear)  $\in$  Functional $\neg$ );
a(rewrite_tac[cleanDirectory_def, graph_thm, functional_def, dir_components,
  get_spec $\Gamma$  MkDirectory $\neg$ ] THEN REPEAT strip_tac THEN asm_rewrite_tac[]);
val cleanDir_fun_thm = save_pop_thm"cleanDir_fun_thm";

```

HOL output

```

cleanDir_fun_thm =  $\vdash$   $\forall$  clear • Graph (cleanDirectory clear)  $\in$  Functional

```

SML

```

| push_goal([],⌈∀ clear • Graph (replaceData clear) ∈ Functional⌋);
| a(rewrite_tac[replaceData_def,graph_thm,functional_def,data_components,get_spec⌈MkData⌋]
|   THEN REPEAT strip_tac THEN asm_rewrite_tac[]);
| val replaceData_fun_thm = save_pop_thm"replaceData_fun_thm";

```

HOL output

```

| replaceData_fun_thm = ⊢ ∀ clear • Graph (replaceData clear) ∈ Functional

```

SML

```

| push_goal([],⌈∀ clear tab • TS_cons tab ∈ Functional
|   ⇒ Fst (cleanColCons clear tab) ∈ Functional⌋);
| a(rewrite_tac[cleanColCons_def,functional_def,>-thm] THEN REPEAT strip_tac);
| a(DROP_NTH_ASM_T 5 (strip_asm_tac o list_∀_elim[⌈x⌋,⌈w⌋,⌈z⌋]));
| val cleanColCons_fun_thm = save_pop_thm"cleanColCons_fun_thm";

```

HOL output

```

| cleanColCons_fun_thm =
| ⊢ ∀ clear tab
|   • TS_cons tab ∈ Functional ⇒ Fst (cleanColCons clear tab) ∈ Functional

```

SML

```

| push_goal([],⌈∀ clear dir • Dir_tables dir ∈ Functional
|   ⇒ Dir_tables (cleanDirectory clear dir) ∈ Functional⌋);
| a(rewrite_tac[cleanDirectory_def,functional_def] THEN REPEAT strip_tac);
| a(POP_ASM_T ante_tac THEN POP_ASM_T ante_tac
|   THEN rewrite_tac[get_spec⌈MkDirectory⌋]);
| a(cases_tac⌈clear dominates Dir_class dir⌋ THEN asm_rewrite_tac[]);
| a(rewrite_tac[r_g-r_thm,graph_thm]);
| a(REPEAT strip_tac);
| a(DROP_NTH_ASM_T 6 (strip_asm_tac o list_∀_elim[⌈x⌋,⌈z'⌋,⌈z''⌋]));
| a(asm_rewrite_tac[]);
| val tables_fun_thm = save_pop_thm"tables_fun_thm";

```

HOL output

```

| tables_fun_thm =
| ⊢ ∀ clear dir
|   • Dir_tables dir ∈ Functional
|     ⇒ Dir_tables (cleanDirectory clear dir) ∈ Functional

```

Now the main result of this section that *hideR* preserves the invariant on states.

SML

```

| push_goal([],⊢∀ clear s • isState s ⇒ isState(hideR(clear,s))⊣);
| a(rewrite_tac[get_spec⊢hideR⊣] THEN REPEAT ∀_tac);
| a(rewrite_tac[get_spec⊢isState⊣,get_spec⊢StateS⊣,↔_def,∩_def] THEN strip_tac);
| a(lemma_tac⊢(s ▷ {dir|clear dominates Dir_exist dir}) % Graph (cleanDirectory clear)
|   ∈ Functional⊣);
| (* *** Goal "1" *** *)
| a(strip_asm_tac(list_∀_elim⊢{dir|clear dominates Dir_exist dir}⊣,⊢s⊣)▷_fun_thm));
| a(strip_asm_tac(∀_elim⊢clear⊣cleanDir_fun_thm));
| a(strip_asm_tac(list_∀_elim⊢(s ▷ {dir|clear dominates Dir_exist dir})⊣,
|   ⊢cleanDirectory clear⊣)%_fun_thm));

```

SML

```

| (* *** Goal "2" *** *)
| a(TOP_ASM_T rewrite_thm_tac);
| a(DROP_NTH_ASM_T 3 ante_tac THEN rewrite_tac[↔_def,get_spec⊢IdeL⊣,
|   get_spec⊢DirectoryS⊣,∩_def,×_def,get_spec⊢Universe⊣,dom_def,
|   ▷_thm,r_%_r_thm,graph_thm,rel_ext_clauses,get_spec⊢$P⊣]);
| a(REPEAT strip_tac);
| a(asm_rewrite_tac[]);
| a(DROP_NTH_ASM_T 4(strip_asm_tac o list_∀_elim⊢x⊣,⊢z⊣));
| a(POP_ASM_T ante_tac THEN rewrite_tac[↔_def,∩_def] THEN strip_tac);
| a(strip_asm_tac(list_∀_elim⊢clear⊣,⊢z⊣)tables_fun_thm));
| a(POP_ASM_T rewrite_thm_tac);
| a(DROP_NTH_ASM_T 2 ante_tac);

```

SML

```

| a(DROP_NTH_ASM_T 2 ante_tac
|   THEN rewrite_tac[cleanDirectory_def,get_spec⊢MkDirectory⊣]
|   THEN cases_tac⊢clear dominates Dir_class z⊣
|   THEN asm_rewrite_tac[↔_def,×_def] THEN strip_tac);
| a(rewrite_tac[∈_in_clauses,rel_ext_clauses,get_spec⊢TableSpecS⊣,
|   get_spec⊢Ide⊣,↔_def,∩_def,r_%_r_thm,graph_thm,cleanTable_def]
|   THEN conv_tac(MAP_C let_conv) THEN strip_tac THEN strip_tac
|   THEN strip_tac THEN strip_tac);
| a(DROP_NTH_ASM_T 3(strip_asm_tac o list_∀_elim⊢x'⊣,⊢z'⊣));
| a(DROP_NTH_ASM_T 4 ante_tac THEN cases_tac⊢clear dominates TS_class z'⊣
|   THEN asm_rewrite_tac[tab_components,get_spec⊢MkTableSpec⊣]
|   THEN strip_tac
|   THEN asm_rewrite_tac[↔_def,×_def,prove_rule[functional_def]⊢{ } ∈ Functional⊣]);

```

SML

```

a(PC_T1 "hol1"rewrite_tac[get_spec⊢ ColCons⊢,get_spec⊢ Num⊢]);
a(strip_asm_tac (list_∇_elim[⊢ clear⊢,⊢ z'⊢]cleanColCons_fun_thm));
a(POP_ASM_T rewrite_thm_tac);
a(∇_tac THEN rewrite_tac[cleanRows_def]);
a(DROP_NTH_ASM_T 7 ante_tac);
a(rewrite_tac[get_spec⊢ RowS⊢]THEN conv_tac(MAP_C let_conv)
  THEN PC_T1 "hol1"rewrite_tac[↔_def,×_def,get_spec⊢ Num⊢,get_spec⊢ DataS⊢,↔_def]);
a(lemma_tac⊢∃ l • TS_rows z' = l⊢THEN_LIST
  [prove_∃_tac,POP_ASM_T rewrite_thm_tac]);

```

SML

```

a(LIST_INDUCTION_T⊢l⊢asm_tac);
(* *** Goal "2.1" *** *)
a(rewrite_tac[⊢_def,map_def,map_null_thm,∈l_def]);
(* *** Goal "2.2" *** *)
a(∇_tac THEN rewrite_tac[map_def,⊢_def,∈l_def]);
a strip_tac;
a(lemma_tac⊢∇ r • r ∈l l ⇒ R_data r ∈ Functional⊢);
(* *** Goal "2.2.1" *** *)
a(REPEAT strip_tac);
a(cases_tac⊢r' = x''⊢);

```

SML

```

(* *** Goal "2.2.1.1" *** *)
a(DROP_NTH_ASM_T 3(ante_tac o  $\forall$ _elim $\lceil$ x'' $\rceil$ ) THEN asm_rewrite_tac[]);
(* *** Goal "2.2.1.2" *** *)
a(DROP_NTH_ASM_T 3(ante_tac o  $\forall$ _elim $\lceil$ r'' $\rceil$ ) THEN asm_rewrite_tac[]);
(* *** Goal "2.2.2" *** *)
a(cases_tac $\lceil$ clear dominates R_exist x'' $\rceil$ THEN asm_rewrite_tac[map_def, $\in$ _def]
  THEN strip_tac );
(* *** Goal "2.2.2.1" *** *)
a(POP_ASM_T rewrite_thm_tac THEN
  rewrite_tac[cleanRow_def,row_components,get_spec $\lceil$ MkRow $\rceil$ ]);
a(lemma_tac $\lceil$ filterRow (Snd (cleanColCons clear z')) (R_data x'')  $\in$  Functional $\rceil$ );
(* *** Goal "2.2.2.1.1" *** *)
a(DROP_NTH_ASM_T 3 (ante_tac o  $\forall$ _elim $\lceil$ x'' $\rceil$ ));
a(rewrite_tac[functional_def] THEN strip_tac);
a(strip_tac THEN rewrite_tac[filterRow_def,cleanColCons_def, $\triangleright$ _thm, $\triangleleft$ _thm,dom_def]);
a(REPEAT strip_tac);
a(DROP_NTH_ASM_T 9 (strip_asm_tac o list_ $\forall$ _elim $\lceil$ x''' $\rceil$ , $\lceil$ w $\rceil$ , $\lceil$ z'' $\rceil$ ));
(* *** Goal "2.2.2.1.2" *** *)
a(strip_asm_tac( $\forall$ _elim $\lceil$ clear $\rceil$ replaceData_fun_thm));
a(strip_asm_tac(list_ $\forall$ _elim $\lceil$ filterRow (Snd (cleanColCons clear z')) (R_data x'') $\rceil$ ,
 $\lceil$ replaceData clear $\rceil$ }_fun_thm));

```

SML

```

(* *** Goal "2.2.2.2" *** *)
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "2.2.2.3" *** *)
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac[]);
val hideR_lemma = save_pop_thm"hideR_lemma";

```

4.4 hide Lemma

We now use the results of sections 4.1, 4.2 and 4.3 to prove that *hide* is a member of *SecureHide*.

SML

```

val secureHide_lemma = save_thm("secureHide_lemma",
  rewrite_rule[secureHideR_lemma,hideR_lemma]hideR_hide_lemma);

```

HOL output

```

secureHide_lemma =  $\vdash$  hide  $\in$  secureHide

```

5 CLOSING DOWN

The following ProofPower instruction restores the previous proof context.

SML

|*pop-pc*();

6 THE THEORY fef010

6.1 Parents

fef009

6.2 Children

fef011 fef040

6.3 Constants

secureHideR $(Class \times (CHAR\ LIST\ LIST \times Directory)\ SET$
 $\rightarrow (CHAR\ LIST\ LIST \times Directory)\ SET)\ SET$

6.4 Definitions

secureHideR $\vdash \forall h$

- $h \in secureHideR$
 $\Leftrightarrow (\forall c_1\ c_2\ s_1\ s_2$
 - $h(c_1, s_1) = h(c_1, s_2) \wedge c_1\ dominates\ c_2$
 $\Rightarrow h(c_2, s_1) = h(c_2, s_2))$

6.5 Theorems

repState_consistent

absState_consistent

$\vdash Consistent$
 $(\lambda (repState', absState')$

- $(\forall a \bullet absState' (repState' a) = a)$
 $\wedge (\forall r$
 - $isState\ r \Rightarrow repState' (absState' r) = r)$
- $(\forall a_1\ a_2$
 $repState' a_1 = repState' a_2 \Leftrightarrow a_1 = a_2)$
- $(\forall r_1\ r_2$
 $isState\ r_1 \wedge isState\ r_2$
 $\Rightarrow (absState' r_1 = absState' r_2$
 $\Leftrightarrow r_1 = r_2))$

- $(\forall s \bullet isState (repState' s))$

dir_components

$\vdash \forall dir_1\ dir_2$

- $dir_1 = dir_2$
 $\Leftrightarrow Dir_tables\ dir_1 = Dir_tables\ dir_2$
 $\wedge Dir_exist\ dir_1 = Dir_exist\ dir_2$
 $\wedge Dir_class\ dir_1 = Dir_class\ dir_2$

tab_components

$\vdash \forall t_1\ t_2$

- $t_1 = t_2$
 - $\Leftrightarrow TS_class\ t_1 = TS_class\ t_2$
 - $\wedge TS_maxRow\ t_1 = TS_maxRow\ t_2$
 - $\wedge TS_colspecs\ t_1 = TS_colspecs\ t_2$
 - $\wedge TS_cons\ t_1 = TS_cons\ t_2$
 - $\wedge TS_rows\ t_1 = TS_rows\ t_2$

row_components

- $\vdash \forall r_1\ r_2$
 - $r_1 = r_2$
 - $\Leftrightarrow R_exist\ r_1 = R_exist\ r_2$
 - $\wedge R_data\ r_1 = R_data\ r_2$

data_components

- $\vdash \forall d_1\ d_2$
 - $d_1 = d_2$
 - $\Leftrightarrow Dat_class\ d_1 = Dat_class\ d_2$
 - $\wedge Dat_item\ d_1 = Dat_item\ d_2$

hideR_hide_lemma

- $\vdash hideR \in secureHideR$
 - $\wedge (\forall\ clear\ s$
 - $isState\ s \Rightarrow isState\ (hideR\ (clear,\ s))$
- $\Rightarrow hide \in secureHide$

replaceData_lemma

- $\vdash \forall c_1\ c_2$
 - $c_1\ dominates\ c_2$
 - $\Rightarrow (\forall d_1\ d_2$
 - $replaceData\ c_1\ d_1 = replaceData\ c_1\ d_2$
 - $\Rightarrow replaceData\ c_2\ d_1 = replaceData\ c_2\ d_2)$

cleanColCons_lemma

- $\vdash \forall c_1\ c_2\ t_1\ t_2$
 - $c_1\ dominates\ c_2$
 - $\wedge cleanColCons\ c_1\ t_1 = cleanColCons\ c_1\ t_2$
 - $\Rightarrow cleanColCons\ c_2\ t_1 = cleanColCons\ c_2\ t_2$

 \subseteq _cleanColCons_lemma

- $\vdash \forall c_1\ c_2\ t$
 - $c_1\ dominates\ c_2$
 - $\Rightarrow Fst\ (cleanColCons\ c_2\ t)$
 - $\subseteq Fst\ (cleanColCons\ c_1\ t)$
 - $\wedge Snd\ (cleanColCons\ c_2\ t)$
 - $\subseteq Snd\ (cleanColCons\ c_1\ t)$

 \subseteq _cleanColCons_lemma1

- $\vdash \forall c_1\ c_2\ t\ col$
 - $c_1\ dominates\ c_2$
 - $\Rightarrow col \in Snd\ (cleanColCons\ c_2\ t)$
 - $\Rightarrow col \in Snd\ (cleanColCons\ c_1\ t)$

cleanRow_lemma

- $\vdash \forall c_1\ c_2$
 - $c_1\ dominates\ c_2$
 - $\Rightarrow (\forall t_1\ t_2\ r_1\ r_2$

$$\begin{aligned}
& \bullet \text{cleanColCons } c_1 \ t_1 = \text{cleanColCons } c_1 \ t_2 \\
& \quad \wedge \text{cleanColCons } c_2 \ t_1 \\
& \quad = \text{cleanColCons } c_2 \ t_2 \\
& \quad \wedge \text{cleanRow} \\
& \quad \quad c_1 \\
& \quad \quad (\text{Snd } (\text{cleanColCons } c_1 \ t_1)) \\
& \quad \quad r_1 \\
& \quad = \text{cleanRow} \\
& \quad \quad c_1 \\
& \quad \quad (\text{Snd } (\text{cleanColCons } c_1 \ t_2)) \\
& \quad \quad r_2 \\
& \Rightarrow \text{cleanRow} \\
& \quad \quad c_2 \\
& \quad \quad (\text{Snd } (\text{cleanColCons } c_2 \ t_1)) \\
& \quad \quad r_1 \\
& = \text{cleanRow} \\
& \quad \quad c_2 \\
& \quad \quad (\text{Snd } (\text{cleanColCons } c_2 \ t_2)) \\
& \quad \quad r_2)
\end{aligned}$$

cleanTable_lemma

$$\begin{aligned}
& \vdash \forall c_1 \ c_2 \\
& \bullet c_1 \text{ dominates } c_2 \\
& \Rightarrow (\forall t_1 \ t_2 \\
& \bullet \text{cleanTable } c_1 \ t_1 = \text{cleanTable } c_1 \ t_2 \\
& \Rightarrow \text{cleanTable } c_2 \ t_1 = \text{cleanTable } c_2 \ t_2)
\end{aligned}$$

cleanDirectory_lemma

$$\begin{aligned}
& \vdash \forall c_1 \ c_2 \\
& \bullet c_1 \text{ dominates } c_2 \\
& \Rightarrow (\forall d_1 \ d_2 \\
& \bullet \text{cleanDirectory } c_1 \ d_1 = \text{cleanDirectory } c_1 \ d_2 \\
& \Rightarrow \text{cleanDirectory } c_2 \ d_1 \\
& = \text{cleanDirectory } c_2 \ d_2)
\end{aligned}$$

 \subseteq _dir_lemma

$$\begin{aligned}
& \vdash \forall c_1 \ c_2 \ s \ x \ y \\
& \bullet c_1 \text{ dominates } c_2 \\
& \Rightarrow (x, y) \in s \triangleright \{\text{dir} | c_2 \text{ dominates } \text{Dir_exist } \text{dir}\} \\
& \Rightarrow (x, y) \in s \triangleright \{\text{dir} | c_1 \text{ dominates } \text{Dir_exist } \text{dir}\}
\end{aligned}$$

dir_class_preserved_lemma

$$\begin{aligned}
& \vdash \forall c \ c_1 \ s \ s_1 \ x \ z \ z_1 \\
& \bullet \text{cleanDirectory } c_1 \ z = \text{cleanDirectory } c_1 \ z_1 \\
& \quad \wedge (x, z) \\
& \quad \in s \triangleright \{\text{dir} | c_1 \text{ dominates } \text{Dir_exist } \text{dir}\} \\
& \quad \wedge (x, z_1) \\
& \quad \in s_1 \triangleright \{\text{dir} | c \text{ dominates } \text{Dir_exist } \text{dir}\} \\
& \Rightarrow (x, z_1) \\
& \quad \in s_1 \triangleright \{\text{dir} | c_1 \text{ dominates } \text{Dir_exist } \text{dir}\}
\end{aligned}$$

secureHideR_lemma

$$\vdash \text{hideR} \in \text{secureHideR}$$

cleanDir_fun_thm

$\vdash \forall \text{clear} \bullet \text{Graph} (\text{cleanDirectory } \text{clear}) \in \text{Functional}$
replaceData_fun_thm
 $\vdash \forall \text{clear} \bullet \text{Graph} (\text{replaceData } \text{clear}) \in \text{Functional}$
cleanColCons_fun_thm
 $\vdash \forall \text{clear } \text{tab}$

- $\text{TS_cons } \text{tab} \in \text{Functional}$
- $\Rightarrow \text{Fst} (\text{cleanColCons } \text{clear } \text{tab}) \in \text{Functional}$

tables_fun_thm
 $\vdash \forall \text{clear } \text{dir}$

- $\text{Dir_tables } \text{dir} \in \text{Functional}$
- $\Rightarrow \text{Dir_tables} (\text{cleanDirectory } \text{clear } \text{dir})$
 $\in \text{Functional}$

hideR_lemma $\vdash \forall \text{clear } s \bullet \text{isState } s \Rightarrow \text{isState} (\text{hideR} (\text{clear}, s))$
secureHide_lemma
 $\vdash \text{hide} \in \text{secureHide}$

7 INDEX

<i>cleanColCons_def</i>	5
<i>cleanColCons_fun_thm</i>	27
<i>cleanColCons_lemma</i>	13
<i>cleanDirectory_def</i>	5
<i>cleanDirectory_lemma</i>	24
<i>cleanDir_fun_thm</i>	26
<i>cleanRows_def</i>	5
<i>cleanRow_def</i>	5
<i>cleanRow_lemma</i>	15
<i>cleanTable_def</i>	5
<i>cleanTable_lemma</i>	22
<i>data_components</i>	8
<i>dir_class_preserved_lemma</i>	25
<i>dir_components</i>	6
<i>dominates_trans</i>	5
<i>fef010</i>	4
<i>filterRow_def</i>	5
<i>hideR_def</i>	5
<i>hideR_hide_lemma</i>	10
<i>hideR_lemma</i>	30
<i>hide_def</i>	5
<i>replaceData_def</i>	5
<i>replaceData_fun_thm</i>	27
<i>replaceData_lemma</i>	11
<i>repState_absState_def</i>	5
<i>row_components</i>	8
<i>secureHideR_lemma</i>	26
<i>secureHideR</i>	9
<i>secureHide_def</i>	5
<i>tables_fun_thm</i>	27
<i>tab_components</i>	7
\subseteq <i>cleanColCons_lemma1</i>	14
\subseteq <i>cleanColCons_lemma</i>	13
\subseteq <i>dir_lemma</i>	24