Datalog Educational System
Release Notes History

Fernando Sáenz-Pérez
Grupo de Programación Declarativa (GPD)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Universidad Complutense de Madrid (UCM)
December, 27th, 2021

Contents

1. Release Notes History

This document lists release notes of all software releases previous to the current one in reverse chronological order. Release notes for the current version are listed in the User Manual.

1.1 Version 6.7 of DES (released on September, 4th, 2021)

- Enhancements:
  - Added string solver for semantic analysis with disequality, inequality and domain constraints
  - Improved precision for tautological and inconsistent semantic warnings involving autocast
  - Added domain constraint to integer solver for semantic analysis
  - Missing join condition, inconsistent condition, constant argument warnings for aggregates
  - Added SQL semantic warning for grouped columns in a **HAVING** clause without an aggregate (Error 18 in [BG06])
  - Added SQL semantic warning for repeated **DISTINCT**
  - Added SQL semantic warning for **COUNT** on a not-null constrained argument
  - Added semantic warning for **OFFSET 0**
  - Dropped semantic false positive for **DISTINCT** in **IN** subqueries, which leads to better compilations
  - Improved SQL semantic analysis precision for automatic type casting
  - An SQL constraint may receive a void name
  - Inform about possible column names when the requested column does not exist
  - An SQL constraint name can be optionally given when altering tables
  - Improved and more precise error messages for SQL grouping statements
  - Interval check in SQL **BETWEEN**
  - Reduced arguments in **ORDER BY** compilations
  - SQL **TOP**, **OFFSET** and **LIMIT** clauses and RA **top** operator can include expressions instead of just integers
  - SQL constants can be specified as 0-arity functions (with parentheses)
  - Simplified compilations for SQL **MINUS DISTINCT** and **INTERSECT DISTINCT**
  - Removed unneeded **DISTINCT**'s in SQL **DIVISION** translation

- Simplification of true goals in `top`
- Reworked help on built-ins, which becomes interactive
- Added help on selection functions
- Added exception for compilation failures
- Added context information in stratum solving for verbose mode
- Added informative message for trying to removing an unexisting constraint
- Formatted exceptions on evaluation error
- More flexible date formats
- Reused temporary rule identifiers
- Simplification of true goals in `top`
- The `/if` command accepts a generic Datalog goal, instead of just a comparison condition. Thus, the condition (goal) may have to be enclosed between parentheses
- The commands `/save_state` and `/restore_state` become TAPI-enabled
- New functions and predicates. For each function below, there is a counterpart Datalog predicate with the same name prepended with `$`, and with an extra final argument as output:
  - `add_months(X,Y)`
    Add to the datetime `X` the number of months `Y`
  - `datetime_add(X,Y)`
    Return the datetime `X` increased by the number `Y`. If `X` is a date, `Y` represents days, and seconds otherwise. This function is equivalent to the overloaded `X + Y`
  - `datetime_sub(X,Y)`
    If `Y` is a number, return the datetime `X` decreased by the days `Y`. If `X` and `Y` are dates, return the number of days between them. If `X` and `Y` are either times or timestamps, return the number of seconds between them. This function is equivalent to the overloaded `X - Y`
  - `greatest(X1,...,Xn)`
    Return the greatest element `Xi` in the lexicographic order
  - `instr(X,Y)`
    Return the first numeric position in the string `X` of the searched substring `Y`
  - `last_day(X)`
    Return the last day of the month for the given datetime `X`
  - `least(X1,...,Xn)`
    Return the least element `Xi` in the lexicographic order
  - `mod(X,Y)`
    `X` modulo `Y`. Apply to two integers and return an integer
  - `nvl2(X,Y,Z)`
    Return either `Y` if `X` is a not null value, or `Z` otherwise
  - `replace(X,Y,Z)`

Replace the string `Y` by `Z` in the given string `X`

- `reverse(X)`
  Reverse the string `X`
- `rpad(X,Y)`
  Return the given string `X` padded to the right with spaces, with the given total length `Y`
- `rpad(X,Y,Z)`
  Return the given string `X` padded to the right with `Z`, with the given total length `Y`
- `to_char(X)`
  Convert a datetime `X` to a string
- `to_char(X,Y)`
  Convert a datetime to a string for a given format
- `to_date(X)`
  Convert the string `X` to a date
- `to_date(X,Y)`
  Convert the string `X` to a date for the given format `Y`
- `trim(X)`
  Remove both leading and trailing spaces from the string `X`

- New commands:
  - `/command_assertions` List commands that can be used as assertions. A Datalog program can contain assertions `:- command(arg1,...,argn)`, where `command` is the command name, and `argi` are its arguments
  - `/input Variable` Wait for a user input (terminated by Intro) to be set on the user variable `Variable`
  - `/list_predicates` List predicates (name and arity). Include intermediate predicates which are a result of compilations if development mode is enabled (cf. the command /development). *TAPI enabled*
  - `/mparse Input` Parse the next input lines as they were directly submitted from the prompt, terminated by a single line containing `$eot`. Return syntax errors and semantic warnings, if present in `Input` (only SQL DQL queries supported up to now). *TAPI enabled*
  - `/mtapi` Process the next input lines by TAPI. It behaves as `/tapi Input`, where `Input` are the lines following `/mtapi` and terminated by a single line containing `$eot`
  - `/parse Input` Parse the input as it was directly submitted from the prompt, avoiding its execution. Return syntax errors and semantic warnings, if present in `Input` (only SQL DQL queries supported up to now). *TAPI enabled*
  - `/run Filename [Parameters]` Reminiscent of old 8 bit computers, this command allows for processing a file but retaining user input for selected commands such as `/input`. Process the

contents of ***Filename*** as if they were typed at the system prompt. Extensions by default are: **.sql** and **.ini**. When looking for a file **f**, the following filenames are checked in this order: **f**, **f.sql**, and **f.ini**. A parameter is a string delimited by either blanks or double quotes (") if the parameter contains a blank. The same is applied to Filename. The value for each parameter is retrieved by the tokens **$parv1$**, **$parv2$**, ... for the first, second, ... parameter, respectively

- **/sandboxed** Synonym for **/host_safe**. Display whether host safe mode is enabled (**on**) or not (**off**). Enabling host safe mode prevents users and applications using DES from accessing the host (typically used to shield the host from outer attacks, hide host information, protect the file system, and so on)
- **/sandboxed *Switch*** Synonym for **/host_safe *Switch***. Enable host safe mode. Once enabled, this mode cannot be disabled

- Changes:
  - The command **/builtins** displays an interactive help based on categories. This can navigate to the interactive command help and vice versa
  - Added separators in interactive help
  - TAPI commands does not output elapsed time
  - String values in SQL results are delimited between single quotes
  - Null values are placed after any other symbol in the lexicographic and numeric orders. This affects the previous ordering of answer displays
  - Write commands changed from category Implementor to Scripting
  - When silent mode is enabled (with the command **/silent on**), display prompt inputs on batch processing are not displayed, thus behaving more similar to applying **/silent *Input*** to each line in the batch file
  - Each call to **rand** returns the same value along fixpoint computation to avoid floundering (though different calls return in general different values)
- Fixed bugs:
  - Exception catching in **/save_ddb**
  - Nulls were not ignored when imposing a foreign key constraint
  - Repeated column names in **INSERT INTO** did not raise an error
  - Exception in some cases of RA natural full join
  - Dropping a table with a foreign key to itselft raised unlimited warnings
  - Some expressions involving SQL statements produced unordered compiled goals
  - Incompatible schemas in SQL/RA division involving subqueries raised an exception instead of informing about the actual error
  - **SELECT ... INTO** modified system variables instead of user variables

- Missing relation renamings in the right operand of SQL **MINUS** and **INTERSECT**
- Tuples causing constraint violations were not always quoted
- The constant **pi** was not detected with its type in SQL expressions
- Translation of nested full outer joins were incorrect
- Incorrect scoping in alias passing for SQL **AND** conditions
- False positive in missing join condition for a conjunctive condition
- Exception in corner cases of SQL autocasting, including arithmetic constants
- Expressions in **IN** statements were not handled properly
- Functions **iif** and **case** were not correctly solved in assumed contexts
- The system flag command_elapsed_time/2 was not deleted for TAPI commands
- Tuples with expressions in the left side of **IN** / **NOT IN** made parsing to fail
- Strings with the name of arithmetic constants were not correctly handled
- Unhandled exception in **group_by** out of bounds for propositional goals
- SQL debugger statistics were not shown with the command **/debug_sql_statistics** after a debugging session
- Exception in run-time autocasting for unquoted atoms
- Some SQL remarks in multiline mode were not correctly parsed
- Some nested **top** calls missed variables and solutions
- Applying **top** on comparison calls dealt no answer
- Arguments of **/debug_dl** were not read properly
- A bug in determining the stratum of a metapredicate goal implied missing answers in some queries

1.2 Version 6.6 of DES (released on January, 25th, 2021)

- Enhancements:
  - TAPI interface for Datalog and SQL debuggers
  - Full Datalog debugger can be applied to the local database (no need for a source file)
  - The original source database is restored after a Datalog debugging session
  - Listed possible answers to Datalog debugger restricted for propositional predicates
  - New **offset** Datalog predicate for pagination
  - Expression evaluation in goal arguments of metapredicates
  - Fractions of seconds are allowed in **time** data type
  - A foreign key can reference the same table it is applied to

- Support for aggregates in **ALL**/**ANY** SQL subqueries
- Improved automatic type casting for SQL built-ins
- Allow for escaped proprietary and standard SQL user delimiters
- Positional arguments allowed for **GROUP BY** and **ORDER BY** clauses, including expressions solved at run time
- Allow for escaped delimiters in SQL user identifiers
- More precise semantic warning message for unneeded argument in SQL **COUNT**
- Added **OFFSET** and **LIMIT** clauses to SQL queries
- Modifier **ALL** added to **INTERSECT** and **EXCEPT** multiset SQL operations
- Added **DEFAULT VALUES** option for **INSERT**
- Added internal error messages for command processing
- The condition in the **/if** command can be surrounded by parentheses
- Uncontrolled exceptions handled in TAPI commands
- New commands:
  - **/breakpoint** Set a breakpoint: start host Prolog debugging
  - **/debug_dl_answer** ***Question Answer*** Answer a question when debugging a Datalog relation. Possible answers are **abort**, **valid**, **nonvalid**, **missing(*Tuple*)**, and **wrong(*Tuple*)**, where ***Tuple*** is of the form **rel(cte$_1$, ..., cte$_n$)**, where **rel** is the relation name and each argument **cte$_i$** is an SQL constant. Placeholders (_) are allowed for missing tuples instead of constants. *TAPI enabled*
  - **/debug_dl_current_question** Display the current question when debuging a Datalog relation. *TAPI enabled*
  - **/debug_dl_explain** Explain the outcome of the last Datalog debugging session. *TAPI enabled*
  - **/debug_dl_node_state** Display Datalog debugging node states. *TAPI enabled*
  - **/debug_dl_set_node** ***Name/Arity State*** Set the state for a node with an unknown state. ***State*** can be either **valid** or **nonvalid**. *TAPI enabled*
  - **/debug_dl_statistics** Display Datalog debugging session statistics. *TAPI enabled*
  - **/debug_sql_answer** ***Question Answer*** Answer a question when debugging an SQL relation. Possible answers are **abort**, **valid**, **nonvalid**, **missing(*Tuple*)**, and **wrong(*Tuple*)**, where ***Tuple*** is of the form **rel(cte$_1$, ..., cte$_n$)**, where **rel** is the relation name and each argument **cte$_i$** is an SQL constant. Placeholders (_) are allowed for missing tuples instead of constants. *TAPI enabled*

  - **/debug_sql_current_question** Display the current question when debuging an SQL view. *TAPI enabled*
  - **/debug_sql_node_state** Display SQL debugging node states. *TAPI enabled*
  - **/debug_sql_set_node *Node State*** Set the state for a node with an unknown state. ***State*** can be either **valid** or **nonvalid**. *TAPI enabled*
  - **/debug_sql_statistics** Display SQL debugging session statistics. *TAPI enabled*

- Changes:
  - Simplified questions in SQL debugger
  - The command interface for **/debug_dl** has been changed to be similar to **/debug_sql**, also adding a new option and optional answer. Additionally, it becomes TAPI enabled
  - Added DBMS-sensitive delimiters in output for relations
  - SQL **EXCEPT DISTINCT** semantics changed for committing to SQL standard
  - The command **/get_relation** is sensitive to the current order answer setting
  - The command **/relation_modified** displays all modified relations
  - The input **/silent *Input*** is not logged
  - Substring for a negative ***Offset*** works as: If ***Length*** is greater than ***-Offset***, then the first ***Length+Offset+1*** characters are returned (up to the length of the string)
  - SQL semantic checking disabled for the commands **/generate_db** and **/debug_sql_bench**

- Fixed bugs:
  - Not found SQL **GROUP BY** column name in aggregate subqueries for comparisons
  - Foreign key constraints were not checked if the asserted rule contained a null
  - Predicate transformation in Datalog debugger generated intermediate user identifiers, possibly clashing with existing predicates
  - Some **TOP** calls were not safe in SQL compilations
  - Some incorrect translations in **ALL** SQL subqueries
  - SQL **INTERSECTION** did not discard duplicates when they were enabled
  - The function **iif** could not be used in an expression
  - Null values were rejected for casting

- Removed incorrect SQL duplicated and constant columns warnings in **`UPDATE`** statements
- Some ill-typed expressions were not identified in the error message
- Labels were not correctly parsed in multiline mode
- SQL user identifiers including blanks were not delimited in some listings
- An SQL statement including an inconsistent condition with type casting enabled was rejected without a hint
- The full Datalog debugger failed for transformed source rules
- SQL strings including a semicolon were not well tokenized
- A TAPI command at the end of a script had to end in a new-line
- Missing answers after solving an embedded implication after the last goal of a previous join
- Trying to **`CREATE OR REPLACE`** a view already defined as a table led to a non-meaningful error message
- Foreign key constraints involving several attributes may fail to be posted
- String constants as arguments of user relations were missed with expressions
- DES exited when a timeout occurred during processing a script file
- Some incorrect compilations with type casting enabled
- Underscored variables as head arguments were removed when consulting a Datalog program
- Plain Datalog debugger did not reset root tuples for statistics
- Datalog basic queries with removed anonymous variable arguments missed the PDG
- The predicate **`order_by`** did not return an answer for a true goal
- Aggregate functions as arguments in expressions might raise an exception
- The Datalog debugger failed when all mutually recursive nodes were non valid
- The command **`/set_timeout`** threw an error instead of informing of an invalid number
- The global timeout was not applied
- The file **`des_pchr.pl`** for the Datalog debugger was not included in binary distros
- Error during parsing exclusive optional modifiers of the command **`/dependent_relations`**
- Error when parsing a restricting query with multiline enabled
- Datalog 0-arity functions **`count`** and **`count_distinct`** were allowed in SQL

- In multiline mode, the terminator ";" did not end some incorrect SQL / RA / TRC / DRC inputs
- Unfolding a negated disjunction was not correct
- SQL `INTERSECT` operation did not discard duplicates when they were enabled
- An aggregate on a single row for a specific program optimization gave a null result
- System facts were listed when development listings were not enabled
- A negated `LIKE` in SQL lead an exception
- Simplifying rules with failing ground primitives lead to error
- Safe goal reordering did not consider the metapredicate `or`

1.3 Version 6.5 of DES (released on April, 10th, 2020)

- Enhancements:
  - Improved automatic type casting for dealing with values of different data types, both at compile-time and at run-time
  - More deterministic type inference, improving analysis time
  - Improved SQL semantic analysis precision for some built-in predicates
  - Better compilations for some aggregate metapredicates, reducing the number of arguments
  - Better SQL compilations by simplifying evaluable goals
  - Better compilations for relations in `group_by` predicate (intermediate predicates with less arguments)
  - Some bad uses of aggregates are issued at compile-time (as errors), instead of at run-time (as an exception)
  - SQL grouping error messages extended with column names
  - Left operand of SQL `LIKE` operator can be a tuple of expressions (instead of just constants or columns)
  - SQL strings can include escaped apostrophes with the backslash, in addition to the already supported double apostrophe
  - The SQL `ALTER TABLE` statement has been extended to deal with table columns: redefining a column (both its data type and constraints) or only changing its data type
  - `ORDER BY` expressions can contain aggregates and include positional notation
  - Added SQL logical operator `XOR`
  - Added `round` function with two arguments, and its corresponding predicate

  - Time and datetime formatting (in addition to the already supported date formatting). Enabled by default and set with the command **/format_datetime**
  - The command **/date_format** admits date formats delimited by single quotes
  - New commands:
    - **/describe *Relation*** Synonym for **/db_schema *Relation***, where ***Relation*** can only be either a table or a view, possibly qualified with the database name (as ***connection:relation***)
    - **/tapi_log** Display whether TAPI logging is enabled. If enabled, both TAPI commands and their results are not logged
    - **/tapi_log *Switch*** Enable or disable TAPI logging (**on** or **off**, resp.) If enabled, both TAPI commands and their results are not logged

- Changes:
  - Changing the type casting mode recompiles views in the local database
  - TAPI inputs and their results are not logged by default
  - Most types from external databases are shown normalized, instead using their particular naming
  - Normalized name types for ODBC connections

- Fixed bugs:
  - A case in simplifying goals was incorrect, dealing to incorrect translations and semantic deductions
  - The **CASCADE** modifier parsing was included in version 6.4 but its processing was not implemented
  - Wrong partial reduction when unfolding some metapredicates for disjunctions
  - A non-valid date format specified by the command **/date_format** would lead to non-termination
  - Incorrect type error when comparing a string constant with a limited-size string type
  - Incorrect exception in SQL truncate function
  - ODBC **varchar** type sizes were not correctly retrieved
  - In SICStus distros, nulls were not grounded for external connections
  - Some SQL error messages about bad groupings were not correct
  - Logging information was duplicated in **/status**

1.4 Version 6.4 of DES (released on January, 20th, 2020)

- Enhancements:

- Dates as strings can be written and displayed in several formats, which are specified with the new command **/date_format *Format***
- Checking for SQL built-in redefinitions
- New functions and predicates. For each function below, there is a counterpart Datalog predicate with the same name prepended with **$**, and with an extra final argument as output:
  - **left(X,Y)**
    Return the first **Y** characters of **X**.
  - **ltrim(X)**
    Remove leading spaces from **X**.
  - **rtrim(X)**
    Remove trailing spaces from **X**.
  - **repeat(X,Y)**
    Repeat the string **X** as many times as **Y**.
  - **right(X,Y)**
    Return the last **Y** characters of **X**.
  - **space(X)**
    Return a string with **X** spaces.
  - **sysdate**
    Synomym for **current_date** (Oracle syntax).
  - **trunc(X,D)** (Truncate)
    Return **X** truncated to **D** decimals.
  - **truncate(X,D)** *ISO*
    Return **X** truncated to **D** decimals.
- The optional **IF EXISTS** clause in dropping tables and views can be placed at the end of the statement
- Cascading drop for tables and views
- New commands:
  - **/date_format** Display the current date format
  - **/date_format *Format*** Set the date format for display and specifying dates as strings. ***Format*** is a string including **YYYY** (numeric year), **MM** (numeric month), **DD** (numeric day of the month), and a (single-char) separator between them. Default is ISO 8601: **YYYY-MM-DD**
  - **/dbs_schemas** Display the schema of each open database: Tables, views and constraints. *TAPI enabled*
  - **/debug_sql_answer *Question Answer*** Answer a question when debugging an SQL view. *TAPI enabled*
  - **/debug_sql_current_question** Display the current question when debugging an SQL view. *TAPI enabled*
  - **/debug_sql_set_node *Node State*** Set the state for a node with unknown state. ***State*** can be either **valid** or **nonvalid**. *TAPI enabled*
  - **/des** Synonym for **/datalog** (switch to Datalog interpreter)

- **/format_datetime** Display whether formatted date and time for datetime values is enabled. It is disabled by default
- **/format_datetime** ***Switch*** Enable or disable formatted date and time for datetime values (**on** or **off**, resp.) If disabled (as default), dates are displayed as **date(year, month, day)**, and time is displayed as **time(hour, minute, second)**, both with positive numbers for each term argument. If enabled, dates are displayed in the date format as specified by the command **/date_format**, and time is displayed as **HH:Mi:SS**.
- **/get_relation** ***Connection Relation*** Display the relation schema and data for the given connection as the Prolog term **schema_data(Schema,Data)**, where **Schema** is of the form **relname(col_1:type_1,...,col_n:type_n)**, **Data** is a list of tuples **relname(val_1,...,val_n)**, **col_i** are column names, **type_i** are type names, and **val_i** are values of the corresponding type **type_i**. If ***Connection***=**$des** and ***Relation***=**answer**, the outcome corresponds to the answer to the last submitted query
- **/keep_answer_table** Display whether keeping the answer table is enabled
- **/keep_answer_table** ***Switch*** Enable or disable keeping the answer table (**on** or **off**, resp.)
- **/relation_modified** Display the relation modified by the last input (Datalog relation, SQL table or view), whether it is typed or not, as the Prolog term **connection_table(Connection, Relation)**, where **Connection** is the connection for which the relation with name **Relation** has been modified. If that last input did not modify any relation, **false** is displayed. *TAPI enabled*

- Changes:
  - Delimiters in messages changed from single to double quotes
  - Dates and times are displayed as formatted strings according to the current date and time formats
  - Display in verbose mode: order of some display messages and some added
  - The predefined **dual** table occurs explicitly in the state of debugging
  - Added new optional arguments and relation specification to the following command:
    - **/dependent_relations [direct] [declared]** ***Relation*** Display a list of relations that depend on ***Relation***. ***Relation*** can be either a pattern **R/A** or a relation **R**. A relation **R** can be either a relation name **N** or **C:N**, where **C** refers to a specific connection and **N** is a relation name. If **direct** is included, the dependency is only direct; otherwise, the dependency is both direct and indirect. If **declared** is included, only declared (typed) relations are included in the outcome. In development mode, system-generated predicates are also considered. *TAPI enabled*

- Fixed bugs:
  - Debugging SQL views with either explicit or implicit references to the **dual** table was incorrect
  - Filenames in consult commands could not contain blanks even when surrounded with double quotes
  - Correlated updates failed
  - Coalesce didn't apply to aggregates
  - Cast raised an exception for some types
  - Some batch SQL updates failed integrity constraints
  - A BC date in an atom was not translated to a non-positive year value
  - A verbose sorting answer message was displayed when no sorting was done. Now, such a message is only issued when the answer is sorted by default
  - Dropping a table with incoming referential integrity constraints did not remove the corresponding Datalog constraints

1.5 Version 6.3 of DES (released on October, 14th, 2019)

- Enhancements:
  - Fuzzy restricted predicates for modifying the semantics of regular predicates by lowering their confidence support
  - Fuzzy intuitionistic logic programming, which allows for assuming new fuzzy clauses, predicates, and fuzzy equations, and for restricting the semantics of existing predicates
  - Checking existency of fuzzy relations for commands, and applicable commands in non-fuzzy mode
  - Fuzzy Datalog rules can be asserted and retracted (formerly, only monotonic DB updates for the **des** expansion were consistent)
  - Proximity equations can be added and removed at any point in both fuzzy expansion modes. The loaded program will be automatically recompiled to reflect changes
  - Added verbose messages when the database is cleared because of changing fuzzy expansion and weak unification algorithm
  - Added some syntax error messages for incorrect fuzzy equations
  - Added **TEXT(*N*)** datatype for supporting MS Access syntax
  - New commands:
    - **/list_undefined** List undefined predicates, i.e., those which are not built-in, not external (ODBC table/view), and not defined with a Datalog rule. *TAPI enabled*
    - **/set** Display each user variable and its corresponding value

  - **`/set Variable`** Display the value for the user variable **`Variable`**

- Changes:
  - Small changes to pretty-printing
  - Asserting a fuzzy equation replaces its older definition
  - Switching between weak unification algorithms does not change transitivity
  - Adjusted PDG display of fuzzy predicate arities (by hiding extra arguments for non-development mode)
  - Fuzzy built-in primitives (such as '**`$t_norm`**') do not longer occur in the PDG
- Fixed bugs:
  - When changing the transitive property of a relation, its closure was not updated
  - Retracting a proximity equation between programs looked for expanded clauses
  - Retracting a proximity equation with no computed closure raised an exception
  - Missing delimiting parentheses when displaying some rules
  - The transitive closure for all of the fuzzy relations used the t-norm for **`~`**
  - The expansion of clauses between similar predicates was not correct for the weak unification algorithm A3
  - Revised messages of fuzzy command in non-fuzzy mode (e.g., **`/t_closure_comp`** raised an error when trying to change its setting)
  - Retracting a fact for a predicate similar to another (with **`/retract`** and **`/abolish Name/Arity`**) did not retract all the compiled rules with fuzzy expansion **`bpl`**

1.6 Version 6.2 of DES (released on January, 21st, 2019)

- Enhancements:
  - Added support for **`ALL`** and **`ANY`** SQL conditions
  - Null handling for string and datetime functions and their corresponding predicates
  - Error messages for non-valid comma-separated sequences
  - The command **`/running_info`** admits the new value **`batch`** for displaying the current processed batch line, when the output is sent only to the log
  - Added function **`exp(expr)`** for calculating the Euler number to the power of **`expr`**, and infix integer division operator **`div`**
  - Non-grouped attributes in the SQL projection list expressions is checked at compile-time rather than raising a run-time exception

- Added the result of SQL compilations in development mode
- The left (sequence)-operand in a **[NOT] IN** SQL operator is allowed to be a (sequence of) expression(s)
- Several standard statements to manage transaction states:
  - **COMMIT [WORK];** Save the current database. *TAPI enabled.*
  - **SAVEPOINT *Name*;** Save the current database to savepoint ***Name***. *TAPI enabled.*
  - **ROLLBACK [WORK] [TO SAVEPOINT *Name*];** Restore database either to the last commit or to the savepoint ***Name***. *TAPI enabled.*
- New SQL functions and corresponding predicates:
  - **coalesce(*List*)** *SQL ISO*
    Return the first non-null value in the list of expressions ***List***.
  - **nvl(*Exp1*,*Exp2*)**
    Return ***Exp2*** if ***Exp1*** evaluates to **null**; otherwise return ***Exp1***.
    Equivalent to **coalesce([*Exp1*,*Exp2*])**.
  - **iif(*Cond*,*Exp1*,*Exp2*)**
    Return ***Exp1*** if ***Cond*** evaluates to **true**; otherwise return ***Exp2***.
  - **nullif(*Exp1*,*Exp2*)**
    Return **null** if ***Exp1*** and ***Exp2*** are equal; otherwise return ***Exp1***.
- New commands:
  - **/csv** Display whether csv dump is enabled. If so, the output csv file name is displayed
  - **/dangling_relations** Display the relations that depend on others which do not exist
  - **/ilog** Display whether immediate logging is enabled. If enabled, each log is closed before user input and opened again afterwards
  - **/ilog *List*** Enable or disable immediate logging. If enabled, each log is closed before user input and opened again afterwards
  - **/restore_ddb** Restore the Datalog database from the default file **des.ddb** . Constraints (type, existence, primary key, candidate key, functional dependency, foreign key, and user-defined) are also restored, if present, from **des.ddb**
  - **/save_ddb** Save the current Datalog database to the file **des.ddb**, rewritting this file if already present. Constraints (type, not nullables, primary key, candidate key, functional dependency, foreign key, and user-defined) are also saved
  - **/set *Variable Expression*** Set the user variable ***Variable*** to the value corresponding to evaluating ***Expression***. An expression can be simply a constant value. Use quotes to delimit a string value (otherwise, it can be interpreted as a variable if it starts with either a capital letter or an underscore). Refer to a user variable by delimiting it with dollars. If a user variable name coincides with the name of a system flag, the system flag overrides the user variable

- Changes:
  - Output is disabled when processing the configuration file, even if it contains a command trying to enable output
  - The substring function and predicate do not raise an exception for offsets and lengths out of limits
- Fixed bugs:
  - Columns in the `USING` list for the right relation were not removed from the output column list
  - Arguments of the command `/time` were not always correctly parsed
  - False positives for ambiguous expression renamings in SQL outer/inner joins
  - The priority of JOIN operators was higher than SELECT in subqueries
  - Some duplicated relation renamings in SQL were not identified
  - The last line of a script was not processed if it was not terminated by a carriage return
  - Evaluating an arithmetic expression containing a `null` value silently failed
  - Exiting the system with a log enabled raised an error
  - `is_null` and `is_not_null` were cached, which dealt to missing tuples
  - Incorrect translation of `IN` operator
  - Incorrect translation in RA queries with duplicated renamed columns
  - Outer joins involving filtered relations could miss tuples
  - Ordering in Datalog assumptions and in `WITH` SQL statements missed displayed solutions
  - Functions with 0 and greater arity might fail to be parsed
  - Some type errors simply listed "Error" with no more information
  - Non-grouped attributes where allowed in the SQL `HAVING` clause
  - An attribute reference to the schema of a nested `USING` clause was not resolved
  - False positive in duplicated relation renamings error
  - False positive for inconsistent condition warning in specific compilations
  - Type inference failed for a CTE including a `GROUP BY` clause
  - The commands `/save_ddb` and `/restore_ddb` failed in presence of integrity constraints
  - Not all nested functions in expressions were extracted
  - Some assumptions were type-rejected
  - Goal ordering for certain unsafe rules was unnecesarilly changed

- A string including only an operator was incorrectly dealt in evaluable expressions with polymorphic functions/operators
- Parsing a **/write** command without an argument failed
- SQL **WHERE** conditions were solved after solving expressions in the **SELECT** list
- SQL infix operators were only allowed in down-case
- False positive in SQL semantic check for some ouput expressions

1.7 Version 6.1 of DES (released on May, 24th, 2018)

- Enhancements:
  - Reworked date and time data type system: Date range extended (since BC 4713 up to the future). Julian and Gregorian calendar support with astronomical Julian Date for calculations.
  - Added conversions between string and date/time data types for automatic type casting and explicit conversions
  - Reworked interactive help on commands
  - New category 'Scripting' for commands
  - More precise error message for unexistent default saved state file
  - Tautological condition SQL check in **SELECT** statements
  - The command **/set_flag** admits an expression instead of just a value to be assigned to a variable
  - Added the clause **INTO** ***SelectTargetList*** for the **SELECT** statement. This allows to communicate SQL return values with the basic scripting system
  - Exposed ODBC errors when figuring out return schemas
  - Stand-alone executables for Ubuntu and Mac OS High Sierra versions compiled with SICStus Prolog, with no dependencies (no need to install other software)
  - Floating point numbers in E-notation accept downcase "**e**" for the base and no longer require a decimal part for the coefficient
  - New commands:
    - **/goto** ***Label*** Set the current script position to the next line where the label ***Label*** is located. A label is defined as a single line starting with a colon (**:**) and followed by its name. If the label is not found, an error is displayed and processing continue with the next script line. This command does not apply to interactive mode
    - **/return** Stop processing of current script, returning a 0 code. This code is stored in the system variable **$return_code$**. Parent scripts continue processing

- **/return *Code*** Stop processing of current script, returning ***Code***. This code is stored in the system variable **$return_code$**. Parent scripts continue processing
- **/set_timeout** Display whether a default timeout is set
- **/set_timeout *Value*** Set the default timeout to ***Value*** (either in seconds as an integer or **off**). If an integer is provided, any input is restricted to be processed for a time period of up to this number of seconds. If the timeout is exceeded, then the execution is stopped as if an exception was raised. If ***Value*** is **off**, the timeout is disabled
- **/stop_batch** Stop batch processing. The last return code is kept. All parent scripts are stopped
- **/time *Input*** Process ***Input*** and display detailed elapsed time. Its output is the same as processing ***Input*** with **/timing detailed**

- Changes:
  - In host safe mode, absolute paths for displaying files are not shown
  - Some commands in the miscellanea category have been turned to be safe on the host safe mode
  - Timeout commands moved to the category 'Timing'
- Fixed bugs:
  - Some HTML formatting in the manual has been fixed
  - Removed extra new line characters in silent mode
  - The command **/restore_state** raised an input processing error
  - Display of SQL conditions involving relations were incorrect in some cases
  - Consulting the DES sources in Unixes versions of SWI-Prolog raised encoding errors
  - Line counting for Datalog metadata was incorrect in SWI-Prolog distros
  - Added help to command **/set_default_parameter**

1.8 Version 6.0 of DES (released on April, 9th, 2018)

- Enhancements:
  - Semantic error checking for SQL statements. Warnings are issued for, among others, the following errors:
    - Either inconsistent, or tautological, or simplifiable conditions
    - Missing join condition
    - Unnecessary join
    - Unused tuple variable
    - Constant output column
    - Duplicated column values
    - Identical tuple variables

    - Comparison with **NULL**
    - **LIKE** without wildcards
- Added a complete algorithm for fuzzy weak unification and a mate resolution algorithm
- Recursion and duplicates are allowed in both TRC and DRC
- Enhanced RA syntax error reporting
- Display of optimized, equivalent SQL statements to RA, DRC, and TRC expressions as a result of compilations to Datalog
- Built-in Datalog predicates for random numbers **'$rand'**/1, which computes a float random number between 0 and 1, and **'$rand'**/2, which computes a random number in its second argument with respect to the 64-bit integer seed in its first argument
- Functions **rand** and **rand(*Seed*)** corresponding respectively, to **'$rand'**/1, and **'$rand'**/2
- Improved Datalog parsing performance
- Further simplifications for relational operators for SQL/RA/TRC/DRC compilations and Datalog when rule simplification is enabled
- Several SQL data types added: **NUMERIC**, **DECIMAL**, and **SMALLINT**
- Revised ACIDE SQL, RA, TRC, DRC and console lexicons highlighting
- Added support for **date**, **time** and **datetime** types for persistent predicates
- **NULL** can be used as a column constraint (which does nothing, indeed)
- Added specific errors for persistent errors
- Revised Spanish translations in ACIDE
- Added support for SQL **NOT LIKE** and **NOT BETWEEN**
- New commands:
    - **/sql_semantic_check** Display whether SQL semantic check is enabled
    - **/sql_semantic_check *Switch*** Enable or disable SQL semantic check **on** or **off**, resp.) When enabled, possible semantic errors are warned
    - **/weak_unification** Display current weak unification algorithm: **a1** (Sessa) or **a3** (Block-based). The algorithm **a3**, though a bit slower at run-time, is complete for proximity relations. However, it shows exponential time complexity for compilation
    - **/weak_unification *Value*** Set the weak unification algorithm: **a1** (Sessa) or **a3** (Block-based). If changed, the database is cleared. The algorithm **a3**, though a bit slower at run-time, is complete for proximity relations. However, it shows exponential time complexity for compilation. Can be used as a directive

- Changes:
  - By default, the console features lexicon highlighting. To disable it, choose `default.xml` (instead of `console.xml`) as its lexicon configuration (either in the contextual menu or in the application menu Configuration → Console → Document Lexicon)
- Fixed bugs:
  - Some SQL functions were not highlighted in ACIDE
  - With duplicates enabled, SQL `EXISTS` and `IN` conditions made their `SELECT` statements to return a tuple for each tuple fulfilling the condition
  - In some cases, unknown attributes in disjunctive SQL `WHERE` conditions
  - Some SQL statements in conditions where not parsed
  - Archive files for Unix distributions contained a couple of files with characters in the extended set that provoked unzip errors
  - Fixed a projection issue in relational algebra
  - Some argument renamings in SQL translations were not displayed
  - Type checking in Datalog to SQL prevented some translations including arithmetical expressions (with the built-in `is`)
  - Listing schemas with the TAPI interface did not follow its protocol
  - Removed an incorrect exception in SQL when type casting was enabled
  - The SQL `LIKE` operator was displayed as the Datalog built-in `'$like'` in SQL listings
  - Integrity constraints were not checked as variants of existing ones, so that some legitimate ones were rejected because of redundancy
  - An error imposing a `CHECK` constraints in a `CREATE TABLE` statement was not displayed
  - Imposing some `CHECK` constraints failed
  - `CHECK` constraints were not typed checked
  - A failing `CREATE TABLE` displayed a `$success` for TAPI
  - An `ADD CONSTRAINT` clause did not display `$success` for TAPI
  - Renaming a table did not correctly change all constraints

1.9 Version 5.0.1 of DES (released on May, 29th, 2017)

- Enhancements:
  - Fuzzy answer subsumption prunes both computation and space by dismissing equal tuples with lesser approximation degrees
  - Fuzzy equations with degree 0 resulting from t-closure are no longer stored
  - PDG and strata updated rather than reconstructed for queries modifying the database (autoviews, SQL queries, ...)

- New syntactic disequality operator **\==**, which does not evaluate its operands by contrast to **\=**
- Added timings to command execution (enabled with the command **/timing on**)
- Added the sizes of both the extensional and call tables to statistics (enabled with **/statistics on** and displayed when verbose output is also enabled with **/verbose on**)
- New commands:
  - **/csv *FileName*** Enables semicolon-separated csv output of answer tuples. If ***FileName*** is **off**, output is disabled. If the file already exists, tuples are appended to the existing file.
  - **/db_rules** Display the number of rules in the database. It includes all the rules that can be listed in development mode (including compilations to core Datalog). Therefore, a number greater than the user rules can be displayed. The system flag **db_rules** is updated each time this command is executed
  - **/display_statistics** Display whether statistics display is enabled
  - **/display_statistics *Switch*** Enable or disable statistics display (**on** or **off**, resp., and disabled by default). Enabling statistics display also enables statistics collection, but disabling statistics display does not disable statistics collection. Statistics include numbers for: Fixpoint iterations, EDB (Extensional Database - Facts) retrievals, IDB (Intensional Database - Rules) retrievals, ET (Extension Table) retrievals, ET lookups, CT (Call Table) lookups, CF (Complete Computations) lookups, ET entries and CT entries. Individual statistics can be displayed in any mode with write commands and system flags (e.g., **/writeln $et_entries$**)
  - **/fuzzy_answer_subsumption** Display whether fuzzy answer subsumption is enabled
  - **/fuzzy_answer_subsumption *Switch*** Enable or disable fuzzy answer subsumption (**on** or **off**, resp. and enabled by default). Enabling fuzzy answer subsumption prunes answers for the same tuple with less approximation degrees, in general saving computations
  - **/list_t_closure *Relation*** List the t-closure of the relation ***Relation*** as fuzzy equations of the form ***X Relation Y=D***, meaning that the symbol ***X*** is similar to the symbol ***Y*** with approximation degree ***D***. Can be used as a directive
  - **/set_var *Var Value*** Set the user variable ***Var*** to ***Value***
  - **/t_closure_entries** Synonym for **/t_closure_entries ~**
  - **/t_closure_entries *Relation*** Display the number of entries in the t-closure of ***Relation***. The system flag **$t_closure_entries$** is updated each time this command is executed

- Changes:

- Enabling statistics does not automatically show statistics unless statistics display is enabled
- Removing the reflexive property does not imply antireflexivity but irreflexivity
- Unquoted output for write family of commands
- Command **/external_pdg** moved to section Dependency Graph and Stratification

- Fixed bugs:
  - PDG and strata were not updated when asserting e-clauses in DES expansion
  - A solve assertion during consulting needs to update PDG and strata
  - The command **/ashell** didn't accept arguments separated by blanks
  - The command **/system** didn't accept general Prolog terms as goals
  - The command **/if** raised an exception
  - Some command arguments were not parsed in uppercase
  - Date and time type error conversions with type casting enabled
  - The test **null=<null** succeeded
  - Some SQL expressions and function arguments were not type checked
  - Optional left parenthesis in **CREATE TABLE** statements surrounding **LIKE** and **AS** specifications were placed incorrectly
  - **CREATE TABLE** ... **AS** was not parsed

1.10 Version 5.0 of DES (released on February, 24th, 2017)

- Enhancements:
  - Fuzzy Datalog as a new query language for fuzzy reasoning. A new system mode FuzzyDES is switched with the new command **/system_mode fuzzy**, which enables fuzzy queries and programs, changing the command prompt to **FDES>**
  - Several commands can be used as assertions so that they are executed if present in consulted Datalog files (the new commands **/solve** and **/system_mode**, **/clear_et**, and most fuzzy commands for portability with Bousi~Prolog)
  - SQL **DEFAULT** constraint for **CREATE TABLE** and **DEFAULT** keyword for **INSERT**
  - Exception control in type casting
  - Several relation primitives supported in the tiny Prolog interpreter
  - New commands:
    - **/external_pdg** Display whether external PDG construction is enabled.

- **/external_pdg** ***Switch*** Enable or disable external PDG construction (**on** or **off**) Some ODBC drivers are so slow that makes external PDG construction impractical. If disabled, tracing and debugging external databases are not possible.
- **/fuzzy_expansion** Display current fuzzy expansion: **bpl** (Bousi~Prolog) or **des** (DES). For each fuzzy equation P~Q=D, the first one generates as many rules for Q as rules for P, whereas for the second one, generates only one rule for Q.
- **/fuzzy_expansion** ***Value*** Set the fuzzy expansion as of the given system: **bpl** (Bousi~Prolog) or **des** (DES). If changed, the database is cleared. The value **bpl** is for experimental purposes and may develop unexpected behaviour when retracting either clauses or equations.
- **/fuzzy_relation** Display each fuzzy relation and its properties.
- **/fuzzy_relation** ***ListOfProperties*** Synonym for **/fuzzy_relation ~** ***ListOfProperties***.
- **/fuzzy_relation** ***Relation ListOfProperties*** Set the relation name with its properties given as a list of: **reflexive**, **symmetric** and **transitive**. If a property is not given, its counter-property is assumed (irreflexive for reflexive, asymmetric for symmetric, and intransitive for transitive).
- **/fuzzy_rel** ***ListOfProperties*** Synonym for **/fuzzy_relation ~** ***ListOfProperties***.
- **/lambda_cut** Display current lambda cut value, a float between 0.0 and 1.0. It defines a threshold for approximation degrees of answers.
- **/lambdacut** Synonym for **/lambda_cut**.
- **/lambda_cut** ***Value*** Set the lambda cut value, a float between 0.0 and 1.0. It defines a threshold for approximation degrees of answers.
- **/lambdacut** *Value* Synonym for **/lambda_cut** ***Value***.
- **/list_fuzzy_equations** List fuzzy equations of the form X~Y=D, meaning that symbol X is similar to symbol Y with approximation degree D. Equivalent to **/list_fuzzy_equations ~**.
- **/list_fuzzy_equations** ***Relation*** List fuzzy equations of the form X Relation Y = D, meaning that predicate X is related under Relation to predicate Y with approximation degree D.
- **/list_t_closure** List the t-closure of the similarity relation **~** as fuzzy equations of the form X~Y=D, meaning that symbol X is similar to predicate Y with approximation degree D. Equivalent to **/list_t_closure ~**.
- **/list_t_closure** ***Relation*** List the t-closure of the similarity relation **~** as fuzzy equations of the form X~Y=D, meaning that symbol X is similar to predicate Y with approximation degree D.
- **/solve** ***Query***
- **/system_mode** Display the current system mode.
- **/system_mode** ***Mode*** Set the system mode to ***Mode*** (**des** or **fuzzy**) Switching between modes abolishes the current database.

  - **/t_closure_comp** Display the way for computing the t-closure, which can be either **datalog** or **prolog**.
  - **/t_closure_comp *Value*** Set the way for computing the t-closure, which can be either **datalog** or **prolog**.
  - **/t_norm** Synonym for **/t_norm ~**.
  - **/t_norm *Value*** Synonym for **/t_norm ~ *Value***.
  - **/t_norm *Relation*** Display the current t-norm for ***Relation***, which can be: **goedel**, **lukasiewicz**, **product**, **hamacher**, **nilpotent**, where **min** is synonymous for **goedel**, and **luka** for **lukasiewicz**.
  - **/t_norm *Relation Value*** Set the current t-norm for ***Relation***, which can be: **goedel**, **lukasiewicz**, **product**, **hamacher**, **nilpotent**, where **min** is synonymous for **goedel**, and **luka** for **lukasiewicz**.
  - **/transitivity** Synonym for **/transitivity ~**.
  - **/transitivity *Value*** Synonym for **/transitivity ~ *Value***.
  - **/transitivity *Relation*** Display the current t-norm for ***Relation***, which can be: **goedel**, **lukasiewicz**, **product**, **hamacher**, **nilpotent**, where **min** is synonymous for **goedel**, and **luka** for **lukasiewicz**.
  - **/transitivity *Relation Value*** Set the current t-norm for ***Relation***, which can be: **goedel**, **lukasiewicz**, **product**, **hamacher**, **nilpotent**, where **min** is synonymous for **goedel**, and **luka** for **lukasiewicz**.

- Fixed bugs:
  - Tracing might deliver incomplete answers for predicates including built-ins
  - Building groups for aggregations failed to build some groups as a result of subsequent iterations of the memo function for recursive predicates
  - Exception raised when either renaming or creating (with the clause **LIKE**) a table for some constraints
  - Arithmetic constants raised an exception in **SELECT** statements
  - 0-degree functions (as **CURRENT_DATE**) were not recognized in **SELECT** statements
  - 0-degree functions and arithmetic constants were not recognized but in lower-case
  - The function **substr** was not recognized by the parser
  - Built-in relations introduced in version 4.2 were not given types. This might deal to incorrect inferred types
  - Trying to insert into a non-existent column generated a second incorrect error message
  - Datetime constants were not normalized in SQL
  - Exception when parsing the command **/set_flag**

- Columns with the name of an arithmetic symbol could not be referenced but with `*`

1.11 Version 4.2 of DES (released on September, 25th, 2016)

- Enhancements:
  - Date and time-related data types `date`, `time` and `datetime` (also `timestamp` following standard SQL) for relations (Datalog and SQL)
  - Evaluable expressions (in both Datalog and SQL) also support string and date/time/datetime types (evaluable expressions were only arithmetic up to previous version)
  - Built-in Datalog predicates for string handling (`$concat`/3, `$length`/2, `$like`/2, `$like`/3, `$lower`/2, `$substr`/4, `$upper`/2)
  - Built-in Datalog predicates for date/time/datetime handling (`$year`/2, `$month`/2, `$day`/2, `$current_date`/1, `$current_time`/1, `$current_timestamp`/1)
  - Functions and operators for string handling (`concat`, `length`, `like-escape`, `lower`, `substr`, `upper`, `||`, `+`)
  - Functions for date/time/datetime handling (`year`, `month`, `day`, `hour`, `minute`, `second`, `current_time`, `current_date`, `current_datetime`).
  - The SQL standard function `extract` is also available (for SQL, RA, TRC and DRC), which extracts the value of the given field from a date/time/timestamp value
  - Overloading for the operators `+`, `-` which can be applied to number, string, and date/time/timestamp types
  - Comparison operators also range on date and time values
  - Time and date value inputs are normalized
  - Datalog and SQL date, time and timestamp constants as, e.g.:
    - `date(2016,08,31)` and `DATE '2016-08-31'`, both representing August 31st 2016 for Datalog and SQL respectively
    - `time(18,30,44)` and `TIME '18:30:44'`, in 24 hour format for the same time for Datalog and SQL respectively
    - `datetime(2016,08,31,18,30,44)` , `DATETIME '2016-08-31 18:30:44'`, `TIMESTAMP '2016-08-31 18:30:44'`, all three representing the same timestamp (first one for Datalog and the others for SQL)
  - Function (`cast`) and predicate (`'$cast'`/3) for type casting between appropriate types
  - Added an ACIDE lexicon for DRC and TRC
  - New SQL DDL statement `CREATE TABLE Name AS Statement`

- Added **trunc** as a synonym for **truncate**

- Fixed bugs:
  - Some exceptions did not clear the asserted answer rules
  - Evaluating a compound arithmetic expression of a variable bound to a null failed for aggregates providing this value

1.12 Version 4.1 of DES (released on April, 19th, 2016)

- Enhancements:
  - A new host safe mode for preventing users and applications using DES from accessing the host (typically used in web applications to shield the host from outer attacks, hide host information, protect the file system, and so on)
  - The whole database (including its current state) can be saved to a file and restored in a subsequent session. An automatic saving and restoring can be stated respectively by adding the commands **/save_state** and **/restore_state** in the files **des.ini** and **des.out**. This way, the user can restart its session in the same state point it was left, including the deductive database, metadata information (types, constraints, SQL text, ...), system settings, all opened external databases and persistent predicates. The command **/autosave [on|toggle|off]** eases this by either adding (**on**) or removing (**off**) these commands from the corresponding files. The option **toggle** toggles its state (from **on** to **off** and vice versa)
  - Added syntax checking for Datalog existential quantifier
  - A renaming for an expression can be used in the SQL **having** condition
  - Added the different operator **!=** (as an alternative to **<>**) to SQL, RA, DRC and TRC conditions
  - Duplicated SQL relation renamings are checked
  - An SQL relation autorenaming is ignored
  - Added alias support to join relations (both inner and outer)
  - Help on a specific command also works for the command name preceded with a slash
  - Help on commands works for commands in any letter-case
  - Help on an unknown entry seeks for alternatives containing the entry as, e.g., **/h show_**, which displays all the commands including **show_**
  - Added the option toggle to the command **/verbose**, which toggles the verbose state
  - New commands:
    - **/autosave** Display whether the database is automatically saved upon exiting and restored upon starting in the file **des.sds** (**on**) or not (**off**).

  - **/autosave *Switch*** Enable or disable automatic saving and restoring of the database (**on** or **off**, resp.) If enabled, the complete database is automatically saved upon exiting and restored upon starting in the file **des.sds**.
  - **/close_dbs** Close all the opened ODBC connections. Make **$des** the current database.
  - **/host_safe** Display whether host safe mode is enabled (**on**) or not (**off**).
  - **/host_safe on** Enable host safe mode. This mode cannot be disabled
  - **/restore_state** Restore the database state from the file **des.sds**. Equivalent to **/restore_state des.sds**, where the current path is the start path.
  - **/restore_state *FileName*** Restore the database state from ***Filename***.
  - **/save_state** Save the current database state to the file **des.sds**. Equivalent to **/save_state force des.sds**, where the current path is the start path.
  - **/save_state [force] *FileName*** Save the current database state to the file ***FileName***. If option **force** is included, no question is asked to the user should the file exists already.

- Changes:
  - The command **/save_ddb** saves the current deductive database in source mode, irrespective of the development mode
  - Type information of views created with SQL are also saved with the command **/save_ddb**
  - The button **verbose** in ACIDE is associated with the command **/verbose toggle**
- Fixed bugs:
  - Variable names were lost during a specific situation during unfolding
  - One TRC syntax error was incorrectly labelled as DRD
  - Simplification flag was not restored upon exceptions
  - Displaying an RA query containing **sort** raised an exception
  - Set difference in both RA and SQL queries might dealt incorrect answers due to a particular case of unfolding
  - Inserting, deleting and updating tuples with SQL were not possible for persistent predicates
  - SQL line remarks in arithmetic expressions were not parsed
  - Tuples with a null in a foreign key were rejected

- Creating a foreign key in SQL did not updated tags for limited domain predicates
- An RA `select` applied to a `project` operation permitted references to attributes not present in the projection list
- Ambiguous column names were not checked in SQL `ON` conditions and relations as nested queries
- The displayed schema for certain natural joins might included incorrect column names
- Displaying of restricting rules for foreign keys was incorrect for a foreign key as a proper subset of table columns
- Parsing external SQL queries failed if the ODBC driver returned extra blanks

1.13 Version 4.0 of DES (released on February, 29th, 2016)

- Enhancements:
  - New formal languages with syntax and safety checking:
    - Domain relational calculus (DRC)
    - Tuple relational calculus (TRC)
  - New Datalog existential quantifier provided with the metapredicate `exists(Vars,Goal)`
  - Improved some SQL syntax error messages
  - Pointing to the location where an unbalanced bracket is found
  - Improved unfolding for more simplified compilations
  - Unsafe calls including a negated goal with unrestricted variables that can be made safe are processed by safety transformations, therefore sugaring syntax for neater formulations
  - Unrestricted, underscored variables are assumed to be existential variables if not occurring in the head
  - Left operand of `is` is checked to be either a variable or number
  - New commands:
    - `/drc` Switch to DRC interpreter (all queries are parsed and executed in DRC).
    - `/drc Query` Trigger DRC evaluation for the query `Query`
    - `/trc` Switch to TRC interpreter (all queries are parsed and executed in TRC).
    - `/trc Query` Trigger TRC evaluation for the query `Query`
- Fixed bugs:
  - RA listings failed due to a non-existent predicate

- An identifier starting with `not` would be split into a `not` metapredicate and the rest of the identifier
- Listings of RA view definitions with `/dbschema` failed (bug introduced in version 3.11)
- Added error message for schema with no column.
- A view with the name and arity of a Datalog built-in could be created.
- Modes were not updated when retracting rules by identifier.
- Negated literals as metapredicate arguments were required to be delimited between parentheses
- Operators with textual names required blanks surrounding them
- Compilation of some logical expressions including `and`, `or` and `not` were not correct due to operator precedence and associativity

1.14 Version 3.12 of DES (released on December, 17th, 2015)

- Enhancements:
  - New ACIDE 0.17 version featuring:
    - Localization to French
    - Multiple editors for new files
    - New Reset and Refresh buttons for the Database panel
    - The Datalog query for Trace Datalog becomes single-lined
    - Keyboard shortcut (F9) for Play
    - Rule (SQL statement, resp.) location becomes disabled by-default in Trace Datalog (Trace SQL, resp.)
    - New Debug SQL panel but with incomplete functionality
    - Bug fixes in several panels (Console, Database, ...)
  - New limited domain predicates, which are predicates where each argument is committed to a finite set of values via referential integrity constraints
  - Restricted predicates can be made persistent
  - New syntax error system. Syntax error messages include the language for which the error has been detected (note that when the selected language is Datalog, then SQL and RA inputs are processed as well). More than one syntax error message is expected for an incorrect input
  - Enhanced syntax error messages for several Datalog built-ins (aggregates, outer joins, metapredicates, ...)
  - Enhanced syntax error messages for SQL
  - Line information for incorrect rules are better formatted
  - Warning messages for (non-recursive) rules which are not delegated to the external DBMS

  - Added context to error syntax messages for unknown SQL and RA identifiers
  - Predefined constraint assertions become TAPI-enabled
  - A new infix metapredicate operator **or** for disjunction tests. Succeeds only once
  - Removed the need for parentheses enclosing an SQL check constraint in a table declaration
  - Predicates resulting from compilations (starting with **$**) can be listed by their system names with **/listing**
  - A few additions to the user manual
  - Formatted ODBC unsupported data type error message
  - New commands:
    - **/close_persistent** If there is only one connection to a persistent predicate, it is closed. Otherwise, the user is warned with the different predicate alternatives. After closing the connection, the predicate is no longer visible except its metadata. The external DBMS keeps its definition. For restoring its visibility again, simply submit an assertion as **:-persistent(*PredSpec*,*DBMS*)**

- Changes:
  - **PCHR.pl** renamed to **des_pchr.pl**
  - Warning messages for recursive rules of persistent predicates become info messages displayed only in verbose mode
  - Verbose message for unrecognized inputs are simplified in one line for all languages, instead of providing a limited help on inputs
  - New file **des_common.pl** for source distributions
- Fixed bugs:
  - Removing either a Datalog or SQL user constraint did not remove the corresponding compiled Datalog rules
  - A compound relation as an argument of **group_by** was not compiled correctly because "non-relevant" arguments were removed, but they do are
  - Tracing might revisit nodes
  - Processing a language command with an argument changed the current language under exceptions
  - Disjunctive conditions in SQL with duplicates repeated tuples fulfilling both disjunctive branches
  - Rules with only built-ins in their bodies did not receive a node in the PDG
  - Along Datalog debugging with **/debug_dl** and development output enabled, some program rules were incorrectly sent to the log
  - Assertions (as, e.g., **:-type**) added via **/tapi** in development mode displayed messages which were not expected in this mode

- A predicate with only facts as defining rules was not correctly located in a stratum (bug introduced in version 3.9)
- After closing a persistent predicate connection, the extension table was not cleared
- Closing a persistent predicate connection did not remove non-delegated rules
- Retracting a compiled rule of a persistent predicate removed only the compilation root
- Wrong singleton warnings for some compiled rules
- Existence error when displaying integrity constraints in database schema under development mode
- Some SQL queries involving nested **UNION**'s were not correctly translated
- An input containing dollars (**$**) might raise an exception
- Compiled ground rules could not be retracted
- Type inference error for numeric types under some particular situations
- A condition argument of a **group_by** including aggregates outside expressions raised an exception
- Body arguments of aggregates could be parsed without parentheses
- Calls to relations with arity 1 and the same name of a Datalog outer join demanded its argument
- The timeout message was not well-located with running info enabled
- Extra carriage returns in ODBC error messages in SWI-Prolog distros

1.15 Version 3.11 of DES (released on August, 5th, 2021)

- Enhancements:
  - Brand new Datalog debugger based on CHR which is started with the command **/debug_dl**
  - SQL debugger oracle automaton provided in the parameter oracle_file (see below)
  - Extended parameters for SQL debugger:
    - **oracle_file(*FileName*)**. The file ***FileName*** includes the database for the intended interpretation
    - **debug([full|plain])**. Perform either a full (including wrong and missing answers) or plain (only **yes** and **no** answers) debugging
    - **order([cardinality|topdown])**. Navigate the computation tree by either looking the smallest relation (**cardinality**) or in a classical top-down (**topdown**) fashion
  - Statistics added to both the Datalog and SQL debuggers

- Datalog and SQL debugging omits questions about nodes in subtrees of valid nodes
- Database instance generator
- **ASSUME** and **WITH** queries can be used in any context a relation is expected. Up to this version, they could only be used as the root query, but not in a nested query
- Extended scripting language
- Access to system flags is provided with the command **/set_flag**
- Stratification and PDG are not computed when closing a connection that is not the current one
- TAPI enabled for **/help** and **/builtins** family of commands
- Simplified extended help on commands: Synonyms are not listed twice
- File names as arguments of commands can be enclosed between double quotes ("), which is useful for file names with trailing or leading blanks
- Help on an incorrect command also shows the preceding slash and its arguments
- Alternative column, table and view names are also hinted for SQL DML and DDL queries and Datalog queries
- Simultaneous logging to different logs is supported. Simply issue as many **/log *File*** commands as needed
- Added the mode **only_to_log** to the command **/output**. This disables output only to the console but not to the log (if enabled)
- Better integration of external processes created with **/shell** as they not require an external window in DES Windows console for SICStus distros
- SQL conditions including **true** and **false** constants are simplified.
- Improved performance when tagging predicates
- New commands:
  - **/batch** Display whether batch mode is enabled. If enabled, batch mode avoids PDG construction
  - **/batch *Switch*** Enable or disable batch mode (**on** or **off**, resp.)
  - **/copy *FromFile* *ToFile*** Synonym for **/cp *FromFile* *ToFile***
  - **/cp *FromFile* *ToFile*** Copy the file ***FromFile*** to ***ToFile***
  - **/current_flag *Flag*** Display the current value of flag ***Flag***
  - **/debug_dl *Name/Arity* *File*** Start the debugger for the relation ***Name/Arity*** which is defined in file ***File***. It is assumed that a predicate name only occurs in the program with the same arity. This debugger implements the framework described in PPDP 2015 paper

- **/debug_sql_bench** ***NbrTables TableSize NbrViews MaxDepth MaxChildren FileName*** With the same parameters as **/generate_db**, generate an SQL database instance and its mutated version. The filename for the first one is appended with **_trust** before the extension.
- **/drop_all_tables** Drop all tables from the current database but **dual** if it exists. If the current connection is an external database, tables in **$des** are not dropped. *TAPI enabled*
- **/drop_all_relations** Drop all relations from the current database but **dual** if it exists. If the current connection is an external database, relations in **$des** are not dropped
- **/drop_all_views** Drop all views from the current database. If the current connection is an external database, views in **$des** are not dropped. *TAPI enabled*
- **/generate_db** ***NbrTables TableSize NbrViews MaxDepth MaxChildren FileName*** These parameters specify the number of tables (***NbrTables***) and its rows (***TableSize***), the maximum number of views (***NbrViews***), the height of the computation tree (i.e., the maximum number of view descendants in a rooted genealogical line) (***MaxDepth***), the maximum number of children for views (***MaxChildren***), and the output filename (***FileName***) for an SQL database instance generation
- **/if** ***Condition Input*** Process ***Input*** if ***Condition*** holds. A condition is written as a Datalog condition, including all the primitive operators and functions
- **/list_relations** List relation (both tables and views) names
- **/list_dbs** Synonym for **/show_dbs**
- **/nolog** ***File*** Disable logging for the given filename ***File***
- **/set_flag** ***Flag Value*** Set the system flag ***Flag*** to ***Value***. Any system flag can be changed but unexpected behaviour can occur if thoughtlessly setting a flag
- **/silent** Display whether silent batch output is either enabled or disabled
- **/silent** ***Option*** Enable or disable silent batch output messages (**on** or **off**, resp.) If this command precedes any other input, it is processed in silent mode (the command is not displayed and some displays are elided, as in particular verbose outputs)
- **/timeout** ***Seconds Input*** Process ***Input*** for a time period of up to the number of seconds specified in ***Seconds***. If the time-out is exceeded, then the execution is stopped as if an exception was raised. Time-out commands can not be nested. In this case, the outermost command is the prevailing one

- Changes:

- Commands are reorganized in help listings
- Warnings are raised for absent parameters in batch processing, instead of raising an exception
- System variables (flags) are replaced in the input, instead of only in the argument of **/write** commands
- When closing a connection, the current one is not changed to **$des**

- Fixed bugs:
  - Removed non-relational predicates which unnecessarily occurred in the Prolog program when debugging a local SQL database
  - A restricted predicate was identified as undefined if no positive rule existed for it
  - Executing a top-N query on a non-terminating view did not terminate
  - Creating a non-standard typed view could raise an exception during type casting
  - Oracle and SQL Server connections for SWI distros did not filter non-user tables
  - The **DUAL** table was created for Oracle
  - Some command synonyms, as **/db_schema**, were not working
  - Command-line parameters for programs invoked by **/shell** were not handled correctly in SWI-Prolog distros
  - Alternative names for unknown objects were not hinted for tables, views, columns and user predicates
  - A schema-less view in a trust file was assumed to be different from an existing view with the same name
  - System calls to compute a Datalog query may result incomplete (e.g., along debugging)
  - Checking the subset question during trusting an SQL specification was wrong
  - An SQL **UNION** statement was incorrectly translated for repeated column names
  - SQL debugging for missing tuples failed sometimes when relations with arguments with the same name were involved in the slicing
  - Some type errors were not shown for **CREATE VIEW** statements
  - Type inference failed for some aggregate expressions when automatic type casting was enabled
  - The **DISTINCT** SQL and **distinct** RA operator did not work for some unfoldings along compilations
  - Formatting error for padding zeroes in displayed milliseconds

- Selecting from an SQL **UNION** did not return in some cases all the tuples due to a parsing issue
- After an SQL query execution, tags were not restored. This might further queries to return unexpected results. Bug introduced in version 3.10

1.16 Version 3.10 of DES (released on January, 21st, 2015)

- Enhancements:
  - Tracing and debugging external relations (via ODBC) is allowed
  - SQL debugging with development mode enabled shows the logic program every time a clause is added
  - The PDG and stratification are also built for external SQL relations when opening an ODBC connection
  - The SQL text which defines an external view is listed for DB2, MySQL, Oracle, and PostgreSQL whenever it is recognized by the DES SQL dialect (other DBMS's might work as well, though not tested)
  - An external view for which there is no a mate Datalog predicate is tagged as an extensional predicate so that only one fixpoint iteration is needed to solve it
  - Extensional database optimization also applies to external relations
  - Incremental building of the PDG which improves performance a bit
  - The equivalent SQL statement to a given RA expression can be inspected by enabling SQL listings with the command **/show_sql on** (The equivalent Datalog rules were possible to show already with the command **/show_compilations on**)
  - Simplified compilations from RA to SQL
  - Relaxed requirement for SQL identifiers: SQL keywords can be used as identifiers for tables, views, and attributes. In order to disambiguate, it might be necessary to enclose the identifier between SQL delimiters
  - A relation with a name which coincide with the name of a Datalog metapredicate with the same arity is rejected
  - Constants can include escaped single quotes
  - Upgraded DDL info messages for external databases
  - Parameterized batch processing: Batch files can contain references to input parameters (**$parv1$**, **$parv2$**, ...)
  - Coloured printed and online manual
  - Updated colors for the console in ACIDE
  - New commands:
    - **/rdg** Display the current relation dependency graph, i.e., the PDG restricted to show only nodes with type information (tables and views) . *TAPI enabled*

  - **`/rdg RelName`** Display the current relation dependency graph restricted to the first relation found with name **`RelName`**. *TAPI enabled*
  - **`/rdg RelName/Arity`** Display the current relation dependency graph restricted to the relation with name **`RelName`** and **`Arity`**. *TAPI enabled*
  - **`/refresh_db`** Refresh local metadata from the current database (either the local, deductive database or an external DB), clear the cache, and recompute the PDG and strata. *TAPI enabled*
  - **`/repeat Number Input`** Repeat **`Input`** as many times as **`Number`**
  - **`/set_default_parameter Index Value`** Set the default value for the i-th parameter (denoted by the number **`Index`**) to **`Value`**

- Changes:
  - System predicates resulting from compilations (starting with **`$`**) only appear in development mode (this applies to consulting the current PDG and strata, and tracing Datalog and local SQL queries)
  - Nodes in the PDG include DDB tables
  - Infix comparisons, built-ins, and MS Access system tables are no longer part of the PDG
  - PDG arcs are ordered by nodes (previously, first were the positive arcs, then the negative arcs)
  - Warning about undefined predicates are only issued when either inserting the offending rule or recomputing the whole PDG
  - System autorenamings are not shown in displayed SQL statements
  - Built-ins in SQL expressions are capitalized in listings
  - Displayed result sets in SQL debugging are ordered
  - Oracle connections are restricted to user tables and views, therefore avoiding to retrieve dozens of system relations
  - All identifiers in system messages are enclosed between single quotes
  - The argument of the command **`/process`** must be enclosed between double quotes (**`"`**) if it contains blanks
  - Calls to recursive relations are not unfolded (both in the translations from SQL and unfolding user rules when **`/unfold on`** is submitted)
  - Deprecated: SQL **`DROP TABLE Tablenames`**
- Fixed bugs:
  - The SQL Debugger tried to slice tables when they were not trusted
  - References to attribute names in the **`GROUP BY`** clause were not always resolved

- SQL listings including the **DIVISION** operator printed its internal representation
- Some metadata for views were retrieved as they were tables for Oracle databases
- Some ODBC diag exceptions were not formatted in the display
- Persistence in DB2 connections was faulty for SWI-Prolog distros
- Listings involving the **top** and **sort** RA operators failed
- The SQL **ON** clause required a blank after it
- Printing expressions in SQL **ORDER BY** failed in listings
- When adding a rule for a new predicate calling a restricted one, the PDG did not include the negative dependency between them

1.17 Version 3.9 of DES (released on November, 26th, 2014)

- Enhancements:
  - Extended SQL **WITH** and **ASSUME** clauses to include negative assumptions
  - Simplified SQL statements sent to external DBMS's
  - Enhanced SQL error messages
  - A circular reference in an expression in the SQL projection list is informed as error unless it is a legal reference (e.g., legal statements as **SELECT a+1 AS a FROM t;** where **a** is an attribute of **t** are allowed)
  - The SQL operator **JOIN** is allowed as a shorthand for **INNER JOIN**
  - More compact compilations for SQL nested outer joins
  - Renamings can be used in the **ORDER BY** SQL clause
  - Controlled SQL and RA renaming errors
  - Syntax error for ambiguous references to columns in both SQL and RA
  - Syntax errors for incorrect SQL and RA renamings (unmatching number of columns and duplicated column names)
  - Revisited scoping in SQL statements and RA expressions
  - Added natural left, right and full outer join RA operators (**nljoin**, **nrjoin** and **nfjoin**, resp.)
  - Added RA sort operator (**sort**)
  - Added RA top operator (**top**)
  - External relations accessed via persistent predicates can be overloaded with the rules in the deductive database via the new metapredicate **st**/1
  - Arguments of Datalog outer joins can be any (possibly compound) goal, instead of a basic goal
  - More precise error syntax for duplicated columns, including their names
  - Control of legal uses of aggregate arguments

- Removed some unnecessary error messages
- Updated syntax colouring for all languages
- Added a new built-in **select_not_null**/3 that selects in its third argument the non-null value in its first and second arguments
- Added explicit modes for several built-in predicates
- New command:
  - **/reset** Synonym for **/restore_default_status**

- Changes:
  - When trying to use a non-open database, a warning is issued and it is automatically opened, instead of asking the user to submit the **/open_db** command
  - Simplified user display of type names for schemas, removing the qualifiers **string**/1 and **number**/1
  - Translations of outer joins use the metapredicate **st**/1 which replaces **lj**/1, **rj**/1 and **fj**/1

- Fixed bugs:
  - Some user-defined constraints failed to detect inconsistent data
  - Dropping a persistent rule raised a non-existing predicate exception
  - Non-terminating persistence assertion when recovering some dependent rules
  - Lack of delimiters in the right arguments of SQL conditions for aliased variables
  - Natural join failed in presence of renamings
  - Natural auto-join failed
  - Natural full outer join provided incorrect results
  - Schema-less RA view definitions failed when compilation display was enabled
  - Renaming in RA did not work for small relation names
  - Inner joins failed for expression arguments
  - Parsing failed for condition-less, non-natural joins in SQL statements
  - Some unfoldings involving calls to negated goals and aliased variables were not correct. This affected to the compilation of SQL and RA queries
  - A **GROUP BY**-less SQL statement was incorrectly translated as a non-aggregate statement
  - Compiled rules as answers from erroneous SQL queries due to unsafe set vars were not removed, therefore enabling unexpected further answers
  - Inferred types of aggregates including expressions were incorrect
  - The SQL debugger raised an exception upon start

- Expressions could not be included in the SQL clause `ORDER BY`

1.18 Version 3.8 of DES (released on July, 30th, 2014)

- Enhancements:
  - Type casting for Datalog fact assertions and SQL insertions and selections. Enabling this with the command `/type_casting on` provides a closer behaviour of SQL statement solving
  - Improved partial reduction and simplifications. This yields to better compilations for SQL and RA statements, and enhances declarativeness of Datalog rules when simplification in enabled
  - Added table renamings for correlated subqueries in `DELETE` and `UPDATE` DML SQL statements
  - The right operand of SQL operator `IN` admits a list of constants, and as an extension, a list of tuples as well. For example: `(1,'a') IN ((1,'a'),(2,'b'))`
  - Datalog rules resulting from SQL and RA compilations include anonymous variables
  - SQL to Datalog compilation displays are affected by the development flag, so that it is possible to inspect compilations after preprocessings with this flag enabled
  - A negative head can include parentheses surrounding the atom
  - Syntax error for incorrect use of a Datalog built-in predicate as an SQL user identifier
  - Slightly better type errors
  - Variable names of persistent rules are kept when restoring a predicate
  - Creating a table with a `CREATE TABLE` statement keeps tuples for this table already stored and checks integrity constraints with respect to these tuples
  - All `CREATE`, `DROP` and `ALTER` SQL statements for external databases do not longer require a preceding `/sql` to be submitted
  - Extended types for persistent metadata
  - Specific code for SQL Server ODBC connections
  - Error messages for rules of persistent predicates including unsupported built-ins
  - Rules including the built-in `is` can be persistent
  - Extended status information with `/status`
  - New version ACIDE 0.16, including upgradings and fixings:
    - Improved Datalog and SQL tracer panels
    - Node contents displayed by double-click

        - Automatic selection and manual location of program rules for trace nodes
        - Better arrangement of graphs
        - Improved Database and Data View panels
        - A graphical interface for database constraints
        - Console, Database and Data View fixes
    - New commands:
        - `/type_casting` Display whether automatic type casting is enabled
        - `/type_casting Switch` Enable or disable singleton warnings (`on` or `off`, resp.)
        - `/close_persistent Name` Close the connection to the persistent predicate `Name`
- Changes:
    - Arithmetic constant and function names are allowed as column names in SQL renamings
    - The PDG and strata are restored instead of recomputed for several SQL statements
    - Creating a table with `CREATE TABLE` checks constraints should a tuple exists in the database already (which could be asserted with a command, for instance)
- Fixed bugs:
    - Several bugs about persistence introduced in previous releases
    - The PDG and strata were not restored after the execution of several DDL, DQL and DML SQL queries
    - SQL Server ODBC connections are now opened in MARS mode, allowing to submit several queries for the same connection
    - Dropping predefined integrity constraints involving several attributes was not always possible
    - The schema for queries to external databases was missing in SICStus distros
    - The SQL text of persistent relations automatically created was not assigned to the correct relation
    - Added infix and prefix to the built-ins help
    - Unsafe rules made its persistence to fail
    - The command `/drop_ic` failed for some constraints involving prefix operators
    - Exception raised during RA `rename` operation
    - Creating a table with the SQL statement `CREATE TABLE LIKE` from a table with attached integrity constraints failed to attach these constraints

- With answer display disabled, the output schema was shown
- View metadata was removed when creating the view including an assumption as a local view definition
- Negative rules cannot be consulted from files
- Topological order was incorrectly computed in some situations
- Type inferencing for a new view in the deductive database failed when there exists the same view name in an open connection
- Combining restricted predicates and hypotheses might lead to extra tuples in the answer
- Database schema display for views containing negative assumptions failed
- The metapredicate **group_by** was not identified in the consequent of the implication, therefore failing its solving
- Parsing of a compound condition as an argument of not required a blank before the opening parenthesis
- Some Datalog type errors for limited-length varying char was not informing about the length
- Selecting from unions involving expressions might lead to missing answers
- Trying to persist a predicate depending on an external relation failed
- Test cases failed to be generated for subqueries including **EXISTS** and **NOT EXISTS**

1.19 Version 3.7 of DES (released on April, 28th, 2014)

- Enhancements:
  - **WITH** and **ASSUME** SQL statements are compiled to Datalog implications. This enables encapsulation and nesting. Local view definitions are no longer global, but local to the statements in the context in which such views are defined
  - Negative assumptions in hypothetical SQL views and queries
  - The same relation can be overloaded as many times as needed in **WITH** and **ASSUME** SQL statements
  - Local view definitions and assumptions can be referred to a relation name instead of its complete schema if the relation exists already
  - Better pretty-print display of rules containing implications
  - Added anonymous variables to compiled code from SQL statements, avoiding annoying messages if queried from Datalog
  - Added redefinition error messages for built-in predicates in assertions
  - TAPI support for:
    - The different versions of the command **/listing**
    - The different versions of the command **/list_et**

- New version ACIDE 0.15, including upgradings and fixings:
  - A graphical tracer for Datalog queries and SQL views
  - A new panel for the asserted database contents
  - Some tweaks to the console configuration
  - Graphical PDG generation in the background
  - Fixed incomplete nodes in the Database panel
  - File editors: Case change and one-click line selection
- New commands:
  - **`/ashell`** ***`Command`*** An asynchronous shell command, i.e., as **`/shell`** ***`Command`*** but without waiting for the process to finish and also eliding output
  - **`/fp_info`** Display whether fixpoint information is to be displayed
  - **`/fp_info`** ***`Switch`*** Enable or disable display of fixpoint information, as the ET entries deduced for the current iteration (**`on`** or **`off`**, resp.)
  - **`/strata`** ***`Name`*** Display the current stratification restricted to predicate with name ***`Name`***.
  - **`/strata`** ***`Name/Arity`*** Display the current stratification restricted to the predicate ***`Name/Arity`***.
  - **`/listing_asserted`** List the Datalog rules that have been asserted with command. Rules from consulted files are not listed. Neither integrity constraints nor SQL views and metadata are displayed. *TAPI enabled.*
  - **`/listing_asserted`** ***`Name`*** List the Datalog rules that have been asserted with command matching ***`Name`***. Neither integrity constraints nor SQL views and metadata are displayed. *TAPI enabled.*
  - **`/listing_asserted`** ***`Name/Arity`*** List the Datalog rules that have been asserted with command matching the pattern ***`Name/Arity`***. Neither integrity constraints nor SQL views and metadata are displayed. *TAPI enabled.*
  - **`/listing_asserted`** ***`Head`*** List the Datalog rules that have been asserted with command whose heads are subsumed by the head ***`Head`***. Neither integrity constraints nor SQL views and metadata are displayed. *TAPI enabled.*
  - **`/list_sources`** ***`Name/Arity`*** List the sources of the Datalog rules matching the pattern ***`Name/Arity`***. *TAPI enabled.*
- Changes:
  - TAPI output for **`/pdg`**, which now also displays nodes (cf. Section 5.17.2)

- Dropped information about local view definitions from **/dbschema** as the whole **WITH** statement is now displayed
- The command **/edit *FileName*** becomes asynchronous
- File **des_persist.pl** becomes **des_persistence.pl**

- Fixed bugs:
  - Tracing did not show the contents of uncached calls
  - The command **/set_editor** without an argument did not work
  - Assuming a type-faulty SQL statement for an existing relation removed its previous schema
  - There were missing answers when computing the consequent of an implication for a non-recursive predicate depending on a recursive one
  - For some from-less SQL systems, the dual table was incorrectly referenced, raising an exception when persisting some rules
  - Asserting a rule for a persistent predicate did not find new predicates that need to be made persistent
  - Some rules including outer joins failed to compile
  - Some issues related to the combination of persistence and implications
  - Dropping the persistence assertion for a predicate with already typed dependent predicates raised an error
  - System predicates resulting from compilations (e.g., from outer joins) were not automatically made persistent
  - A typo in type inferencing avoided to infer some types
  - Translating a Datalog rule to SQL with repeated variables in the head was incorrect
  - Some extra types along type inferencing were lost
  - Field names and types were lost when renaming a view by using the SQL statement **RENAME VIEW**

1.20 Version 3.6 of DES (released on March, 11th, 2014)

- Enhancements:
  - Restricted predicates, which are regular predicates including restrict rules. A restrict rule defines the tuples which are dropped from the meaning of its predicate
  - Negative assumptions in hypothetical Datalog rules and goals
  - Dynamic building of the PDG and strata along hypothetical computations, including extended verbose info for assumed rules, PDG, strata and contexts
  - New version ACIDE 0.14, including upgradings and fixings to:
    - Localization

        - File Editors
        - Data View panel
        - PDG panel
    - Added **BETWEEN** SQL operator for **WHERE** and **HAVING** conditions
    - Multiset expressions in SQL select statements and conditions
    - Added syntax support for HR-SQL
    - Added SQL **FLOAT** as an alias to the already supported **REAL** type
    - Reordered subgoals in metapredicate arguments for efficiency
    - Program unfolding for rules resulting of a rule compilation
    - Expression simplification extended to all comparison operators (when simplification is enabled with command)
    - Better simplifications
    - New commands:
        - **/reorder_goals** Display whether pushing equalities to the left is enabled
        - **/reorder_goals** ***Switch*** Enable or disable pushing equalities to the left (**on** or **off**, resp.)
        - **/unfold** Display whether program unfolding is enabled
        - **/unfold** ***Switch*** Enable or disable program unfolding (**on** or **off**, resp.) Unfolding affects to the set of rules which result from the compilation of a single source rule. Unfolding is always forced for SQL and RA compilations, irrespective of this setting
- Changes:
    - Negation **not**/1 is redefined as a prefix operator. Older syntax is straightforwardly supported
- Fixed bugs:
    - Some expressions including SQL queries in conditions failed to compile
    - Group-by goal arguments were not unfolded in SQL-to-Datalog translations
    - Top-N queries were incomplete for several uses of the same relation along a computation
    - Unfolding of recursive rules were incorrect
    - Simplification of a disjunction with a true goal was incorrect

1.21 Version 3.5 of DES (released on February, 3rd, 2014)

- Enhancements:
    - New version ACIDE 0.13, including upgradings and fixings to:
        - File Editors

        - Data View panel
        - PDG panel
        - Relocatable panels
    - Added **ALTER TABLE** SQL statement for adding and dropping table constraints
    - Singleton variable warning added for both syntactic and semantic singletons
    - Better compilation of aggregate goals, avoiding unnecessary columns
    - A configuration file **des.cnf** can be added with inputs to be silently and batch processed
    - Batch processing can be nested
    - Input history enabled for SWI distros, allowing to recover input from former sessions
    - Added options **'no'** and **'prolog'** to the command **/prompt**
    - New commands:
        - **/display_banner** Display whether the system banner is displayed at startup
        - **/display_banner *Switch*** Enable or disable the display of the banner at startup (**on** or **off**, resp.)
        - **/safety_warnings** Display whether safety warnings are enabled
        - **/safety_warnings *Switch*** Enable or disable safety warnings (**on** or **off**, resp.)
        - **/singleton_warnings** Display whether singleton warnings are enabled
        - **/singleton_warnings *Switch*** Enable or disable singleton warnings (**on** or **off**, resp.)
        - **/undef_pred_warnings** Display whether undefined predicate warnings are enabled
        - **/undef_pred_warnings *Switch*** Enable or disable undefined predicate warnings (**on** or **off**, resp.)
- Changes:
    - Info messages for unsafe, set, and unrestricted variables span only one line
    - Singleton variables appearing along compilations are named as anonymous
    - Changed 'Undeclared' by 'Undefined' for undefined predicate warnings in queries
    - Slight changes is several info, warning and error messages
    - Rearranged help on commands, adding new categories

- Relaxed the requirement for parentheses surrounding the parameter of the SQL `CHECK` clause

- Fixed bugs:
  - A group by operation was not terminating when providing fresh nulls as in outer joins
  - Some anonymous variables were incorrectly removed upon some compilations
  - Variable names lost in the translation of full joins (for compiled rules as seen in development mode)
  - Not all undefined predicates were warned in queries
  - Some basic compound goals in aggregates were not translated
  - Added (extra) input modes for rules with set variables in head
  - Missing blank trailing line after some syntax errors with compact listings disabled
  - Some state flags were not restored upon exceptions
  - Forcing DES to compute SQL statements for an opened ODBC database failed for assumptions and local view definitions
  - Changed compilation message for RA statements that referred to SQL instead of RA
  - RA statements could not be issued from opened ODBC connections for defining new relations
  - Fixed documentation on command `/statistics`

1.22 Version 3.4 of DES (released on December, 18th, 2013)

- Enhancements:
  - New version of ACIDE 0.12, including as upgrades:
    - New PDG panel, displayed either when a `/pdg` command is issued or on demand in the View menu. Display properties of this panel have been added to the application menu
    - Relocatable panels. A panel can be dragged to another panel position, interchanging both panels' locations
  - RA extended projection includes also the possibility of renamings
  - Added listing of offending rules for set variables unsafety
  - Added mode information for set variables unsafety
  - Creation of incorrect views no longer displays the generic error message
  - Some incorrect SQL statements due to aggregates are catched
  - Constraints are checked along updates
  - Datalog division operator can be nested

- Changes:
  - Submitting an order_by related query overrides current `/order_answer` flag. There is no need to submit `/order_answer off` to make sorted listings work
  - Duplicate elimination (duplicates disabled and distinct operator) considers different nulls as the same null, therefore behaving more closely to SQL
  - Variable order in compiled rules
- Fixed bugs:
  - The division operation was incorrect in RA and SQL
  - Aggregates included extra tuples along nested hypothetical assumptions
  - Removed undeclared predicate warning for `true`/0 when occurring as a condition
  - The predicate `group_by` was functioning incorrectly in some cases, depending on previous tabled results
  - Undefined predicate warning removed for `false`/0
  - SQL `NOT EXISTS` clause was incorrectly translated by adding a top-1 goal
  - Computing the `order_by` predicate was omitted for `/order_answer on`. So, a query as `top(1,order_by(t(X),[X]))` might deliver an incorrect answer
  - In multiline mode, when processing a file containing a tab and an SQL line remark (`--`), then the next line was considered as a part of the remark
  - Built-ins were tried to be made persistent for persistent rules including them
  - Retracting an IDB rule for a persistent predicate did not refresh the database schema
  - Assuming IDB rules in an hypothetical implication lead to exceptions in some cases
  - The combination of top and distinct metapredicates produced extra tuples
  - Translation of the Datalog division operator lead to remove head variables not involved in the operation
  - Anonymous variables in consulted rules were given names

1.23 Version 3.3.2 of DES (released on October, 22nd, 2013)

- Enhancements:
  - Rule-level modes, allowing to update modes on rule retracting
  - Adding, removal and updating a mode assertion are reported in verbose mode
  - Simplified and more efficient computation of aggregate predicates

- Arguments of aggregate functions and predicates can be arithmetic expressions, as in `sum(t(X,Y),X*Y,S)`
- SQL aggregate arguments can also be arithmetic expressions, as in `SELECT SUM(a*b) FROM t`
- Added warnings on dangling relations when dropping tables and views
- Display of the offending constraints due to syntax errors when consulting files
- Clause `IF EXISTS` added to SQL statement `DROP VIEW`
- Display of view name alternatives for `DROP VIEW`
- New command:
  - `/use_ddb` Shorthand for `/use_db $des`

- Changes:
  - Meanings of subqueries are displayed in order in the Datalog debugger
  - Dropping a persistent assertion for which further persistent assertions have been made automatically drops all such assertions too
  - Assumptions in hypothetical rules for different contexts are considered different when duplicates are enabled
- Fixed bugs:
  - Modes were not listed for all unsafe rules
  - Syntax errors when reading a file lead to an exception
  - TAPI outputs were added both with a leading and a trailing carriage return when compact listing was disabled
  - Fixed length types raised an exception when persisting a predicate
  - A persistent predicate overloaded with a local definition in a `WITH` SQL clause caused its external view to be dropped
  - String data types were not handled correctly for persistent predicates, raising errors and/or exceptions
  - The division operation was incorrectly translated in some cases

1.24 Version 3.3.1 of DES (released on July, 28th, 2013)

- Enhancements:
  - No longer need to load a file for debugging Datalog goals
  - Less memory requirements for debugging Datalog goals
  - Expected modes for predicates with unsafe rules are automatically added
  - Some dead code removal
  - New commands:

- **/restore_default_status** Restore the status of the system to the initial status, i.e., sets all user-configurable flags to their initial values, including the default database and the start-up directory
- **/list_modes** List the expected modes for unsafe predicates in order to be correctly computed. Modes can be 'i' (for an input argument) and 'o' (for an output argument)
- **/list_modes *Name*** List expected modes, if any, for predicates with name ***Name*** in order to be correctly computed
- **/list_modes *Name/Arity*** List expected modes, if any, for the given predicate ***Name/Arity*** in order to be correctly computed

  - New port to SWI-Prolog 6.4.1

- Changes:
  - Fixpoint iterations displayed in statistics show the number of iterations in **solve_star** (*solve** [Diet87])
  - Disabled EDB optimization
  - Current file system path is not displayed when changing it unless verbose mode is enabled
  - Unified predicate existence message for **/debug_datalog**
  - Added current database info to **/status**
- Fixed bugs:
  - Incorrect message display when trying to successively disable a multi-value flag
  - Uppercase letters in the extended set of symbols were not parsed
  - Non-recursive predicate optimization was not compatible with Datalog debugging

1.25 Version 3.3 of DES (released on July, 12th, 2013)

- Enhancements:
  - ACIDE bundle includes a new version 0.11 featuring a database panel for managing both DES databases (default **$des** and external databases) and regular ODBC connections (tested with MySQL and DB2). In addition, it has been enhanced w.r.t. last bundle (including version 0.9) and fixes several bugs
  - Nulls can be disabled for trying pure Datalog. Also, disabling nulls enhances performance a bit (most likely, not noticeable)
  - Upgraded performance. Some tweaks for performance optimization
  - Extended support for negation in hypothetical rules
  - Help listings (commands and built-ins) restricted to a limited width
  - Added new ISO built-in infix operator **mod**

- Redefinition of built-in comparison operators are avoided
- New commands:
  - **/display_nbr_of_tuples** Display whether display of the number of computed tuples is enabled
  - **/display_nbr_of_tuples *Switch*** Enable or disable display of the number of computed tuples (**on** or **off**, resp.)
  - **/nulls** Display whether nulls are enabled
  - **/nulls *Switch*** Enable or disable nulls (**on** or **off**, resp.) If nulls are disabled, calls to outer join predicates included in already-loaded rules will fail, and attempts to use outer joins will not succeed. This, coupled with **/duplicates off** (as by default) allows to play with pure Datalog with negation and arithmetic built-ins
  - **/optimize_cc** Display whether complete computations optimization is enabled
  - **/optimize_cc *Switch*** Enable or disable complete computations optimization (**on** or **off**, resp. and enabled by default) Fixpoint iterations and/or extensional database retrievals might been saved
  - **/optimize_ep** Display whether extensional predicates optimization is enabled
  - **/optimize_ep *Switch*** Enable or disable extensional predicates optimization (**on** or **off**, resp. and enabled by default). Fixpoint iterations and extensional database retrievals are saved for extensional predicates as a single linear fetching is performed for computing them
  - **/optimize_edb** Display whether extensional database optimization is enabled
  - **/optimize_edb *Switch*** Enable or disable extensional database optimization (**on** or **off**, resp. and enabled by default). Extensional database retrievals are saved for the extensional part of the deductive database
  - **/optimize_nrp** Display whether non-recursive predicates optimization is enabled
  - **/optimize_nrp *Switch*** Enable or disable non-recursive predicates optimization (**on** or **off**, resp. and enabled by default). Memoing is only performed for top-level goals
  - **/optimize_st** Display whether stratum optimization is enabled
  - **/optimize_st *Switch*** Enable or disable stratum optimization (**on** or **off**, resp. and enabled by default). Extensional table lookups are saved for non-recursive predicates calling to recursive ones, but more tuples might be computed if the non-recursive call is filtered, as in this case an open call is submitted instead (i.e., not filtered)

  - **/write *String*** Write ***String*** to console. ***String*** can contain system variables as **$stopwatch$** (which holds the current stopwatch time) and **$total_elapsed_time$** (which holds the last total elapsed time) (See Subsection 5.13.11.1 for system variables)
  - **/writeln *String*** As **/write** but adding a new line at the end of the string
  - **/write_to_file *File String*** Write ***String*** to ***File***. If ***File*** does not exist, it is created; otherwise, previous contents are not deleted and ***String*** is simply appended to ***File***. ***String*** can contain system variables as **$stopwatch$** (which holds the current stopwatch value) and **$total_elapsed_time$** (which holds the last total elapsed time)
  - **/writeln_to_file *File*** As **/write_to_file** but adding a new line

- Changes:
  - Behaviour of **/compact_listings *Switch*** is immediate (neither trailing blank line when enabling compact listings nor absent blank line when disabling)
  - SQL-to-Datalog compilations for the division are not displayed unless development listings are enabled, analogously to outer join operations
  - Operator **\/** in hypothetical literals has been changed to the more appropriate **/**
- Fixed bugs:
  - During computing implications, some rules were not memorized correctly
  - Assumed rules in hypothetical queries and rules were not tested for safety
  - Parsing of **/shell** failed for arguments containing a comma. Now, the characters following the command are directly sent to the shell
  - Attributes in where conditions were not parsed in the command **/des**
  - External relations were warned as non-existing when processing a SQL statement with the command **/des**
  - Some commands did not accept upper case switches
  - Closing an ODBC connection broke external metadata retrieval for subsequent connections
  - The command **/listing** duplicated the rules for persistent predicates
  - The command **/rm *File*** (with synonym **/del**) did not find the file to remove for SICStus distros
  - Missing display of top-level query for exploded queries in normal listings
  - When **show_sql** was enabled, some tasks related to persistent predicates failed, as retracting rules

- Hypothetical Datalog queries for persistent predicates did'nt retract assumed rules and facts

1.26 Version 3.2 of DES (released on February, 2nd, 2013)

- Enhancements:
  - Datalog hypothetical queries and rules. The implication operator **=>** is added to support this
  - Datalog division operator. This operator implements relational division but without resorting to schemas. Instead, variables are used
  - SQL **CHECK** clause as both column and table constraints, including any SQL check as occurring in a **WHERE** clause
  - Functional dependencies can be specified in **CREATE TABLE** SQL statements, *à la* DB2, with the **CHECK** constraint **DETERMINED BY**
  - New Datalog metapredicates **order_by**/2 and **order_by**/3, which allow multi-key sorting on answers
  - SQL **ORDER BY** clause is now working and with multi-key sorting
  - SQL **INSERT INTO** … **VALUES** allows a sequence of tuples to be inserted
  - New commands:
    - **/order_answer** Display whether displayed answers are ordered by default
    - **/order_answer** ***Switch*** Enable or disable a default (ascending) ordering of displayed computed tuples (**on** or **off**, resp.)
    - **/statistics** ***Keyword*** Display statistics for ***Keyword*** (**runtime** or **total_runtime**). For **runtime**, this command displays the CPU time used while executing, excluding time spent in memory management tasks or in system calls. For **total_runtime**, this command displays the total CPU time used while executing, including memory management tasks such as garbage collection but excluding system calls.
  - Null identifiers in user queries are reused when possible
  - Non existing columns are reported when asserting Datalog predefined constraints
  - Added postfix arithmetic built-in **+**
  - Enhanced parsing of SQL, RA and Datalog arithmetic expressions
  - New port to SWI-Prolog 6.2.6
- Changes:
  - Body output of the command **/status** is ordered
  - For **/dbschema** output, a Datalog integrity constraint is displayed under a table if it only refers to this table and under the (database) integrity constraints otherwise. Therefore, Datalog constraints are not listed more

than once as in previous versions, in which a constraint was listed for each table it referred to. If a constraint is created with a **CREATE TABLE** ***Tablename*** statement, it is listed under the table ***Tablename*** even when it refers to other tables or views

- ISO Prolog **xor** bitwise exclusive or replaces **#** operator

- Fixed bugs:
  - Asserting a fact did not update the predicate dependency graph always
  - The metapredicates **top** and **distinct** raised unsafe rule warnings on safe rules in some cases
  - Only one not null column constraint was stored for a **CREATE TABLE** statement
  - Grouping might not create duplicates for aggregation results as in: **group_by(t(_X,Y),[_X],S=sum(Y))**

1.27 Version 3.1 of DES (released on December, 20th, 2012)

- Enhancements:
  - Deductive engine inference is now possible for an open ODBC connection, enabling to directly submit queries which are not supported by the external DBMS (cf. **/des_sql_solving** command)
  - Division operation is now supported for the relational algebra query language
  - SQL has been extended beyond standard with the relational division operation as an operator in the **FROM** clause of a **SELECT** statement
  - New commands:
    - **/db_schema** Synonym for all variants of **/dbschema**
    - **/des** ***Input*** Force DES to solve ***Input***. If ***Input*** is a SQL query, DES solves it instead of relying on external DBMS solving. This allows to try the more expressive queries which are available in DES (as, e.g., hypothetical and non-linear recursive queries)
    - **/des_sql_solving** Display whether DES is forced to solve SQL queries for external DBs
    - **/des_sql_solving** ***Switch*** Enable or disable DES SQL solving for external DBs
    - **/edit** ***Filename*** Edit ***Filename*** by calling the predefined external text editor. This editor is set with the command **/set_editor**
    - **/format_timing** Display whether formatted timing is enabled
    - **/format_timing** ***Switch*** Enable or disable formatted timing (on or off, resp.)
    - **/prompt** Display the prompt format

        - **/prompt *Switch*** Set the format of the prompt. The value **'des'** sets the prompt to **'DES>'**. The value **'des_db'** adds the current database name DB as **'DES:DB>'**. Finally, **'plain'** sets the prompt to '>'. Note that, in any case, if a language other than Datalog is selected, the language name is also displayed before ">"
        - **/set_editor** Display the current external text editor
        - **/set_editor *Editor*** Set the current external text editor to ***Editor***
    - Stopwatch precision of 1 millisecond for SICStus distros
    - Better performance for persistent predicates updates by identifying base cases (facts)
    - Better SQL and RA compilations to Datalog
    - SQL to Datalog compilations are also displayed for DML statements
    - Added verbose info on solving SQL and RA queries
    - Better identification with relation names of non-existing columns
    - Added the SQL type **NUMBER(POS,DEC)**, although not checked as domain restrictor
    - Support for persistency tested in and adapted for more external RDBMS's (IBM DB2 10.1, Sybase ASE 15.7, Oracle 11g in addition to the already tested MS Access 2003, MySQL 5.1, PostgreSQL 9.1, and MS SQL Server 2008)
    - New port to SICStus Prolog 4.2.3
    - New port to SWI-Prolog 6.2.4
    - More examples (e.g., puzzle **bridge.pl** and **even_odd.sql**)
- Changes:
    - As the ODBC connection to Oracle RDBMS is really slow in some systems, for opening a connection, only user tables and views are read to build the PDG
    - When an opened ODBC connection is the current one, queries are no longer parsed by DES
    - Output display disabling also affects to logging
    - Tracing a view also shows its definition
    - Variables as shown in type errors become named
    - Visibility rules of RA are as original proposal
    - Last stage of source-to-source transformations is now displayed with **/show_compilations on**
- Fixed bugs:
    - The command **/abolish** with arguments and referring to a non-persisted predicate raised an input processing error

- Mutually recursive definitions in a single SQL statement led to compilation rejection
- Query schema was missing in the query outcome (bug introduced in version 3.0)
- Some running information were displayed when output was off
- Some non-ground arithmetical expressions generated an exception during parsing
- Datalog rules as compiled from a SQL query were not removed upon an exception
- In SICStus distros, SQL answers with repeated column names were not computed correctly for an ODBC connection
- `count(*)` was simply displayed as `count` in SQL statements for command `/dbschema`
- The full join operation returned more tuples than expected
- Some SQL keywords (e.g., `IN`) required a leading or a preceding blank (an opening and closing parenthesis, resp., is also allowed now)
- `IS NULL` and `IS NOT NULL` SQL conditions were not parsed (bug introduced in version 2.7)
- Rules and SQL statements including null checking were not tested for safety
- 'Group by' statements and rules including predicates/relations as data sources in the having condition were not correctly computed because the predicate dependency graph didn't include such data sources
- Some safe rules including the predicate `group_by` were detected as unsafe
- Some code simplifications involving the metapredicate `distinct` were incomplete
- After an exception during solving a SQL query, compilations were not removed
- Some translations of SQL statements involving `NOT EXIST` clauses were not correct
- The condition in the predicate `group_by` was not tested for safety so that no program transformations were applied to possibly develop safe rules
- Inserting a foreign key assertion was not checked correctly in all cases
- Inserting a tuple into a table with a compound primary key was not checked correctly in all cases
- Some Datalog simplifications forced by SQL compilations were wrong
- A nested `UNION` (removing duplicates: `DISTINCT`) SQL statement lost the connecting references
- Some expression and column references in SQL statements have been fixed
- In some cases, a type error relating matching numeric expressions was raised

1.28 Version 3.0 of DES (released on May, 10th, 2012)

- Enhancements:
  - New commands:
    - **`/close_db Name`** Close the given ODBC connection. *TAPI enabled*
    - **`/drop_assertion`** Drop an assertion
    - **`/start_stopwatch`** Start stopwatch. Precision depends on host Prolog system (1 second or milliseconds)
    - **`/stop_stopwatch`** Stop stopwatch
    - **`/reset_stopwatch`** Reset stopwatch
    - **`/display_stopwatch`** Display stopwatch
    - **`/list_persisted`** Display the persisted predicates. *TAPI enabled*
    - **`/show_dbs`** Display the open database connections. *TAPI enabled*
    - **`/show_sql`** Display whether SQL statements which are sent to an external database are to be displayed
    - **`/show_sql Switch`** Enable or disable display of SQL statements which are sent to an external database (**`on`** or **`off`**, resp.)
    - **`/use_db Name`** Make **`Name`** the current ODBC connection. *TAPI enabled*
    - **`/dbschema Connection:Name`** Display the database schema for the given view or table name in the given connection
    - **`/license`** Display GPL and LGPL licenses. If not found, please visit http://www.gnu.org/licenses
  - New assertions:
    - **`:-persistent(PredSpec[,Connection]))`** Make a predicate to persist on an external RDBMS via an ODBC connection. **`PredSpec`** can be either the pattern **`PredName/Arity`** or **`PredName(Schema)`**, where **`Schema`** can be either **`ArgName1, …, ArgNameN`** or **`ArgName1:Type1, …, ArgNameN:TypeN`**. If a connection name is not provided, the current open database is used
  - Binary flags in commands are no longer case-sensitive
  - New port to SICStus Prolog 4.2.1. This release fixes in particular some issues with ODBC connections (exceptions about misencoded string in non-ASCII ODBC messages, and incorrect handling of SQL_BIGINT and related types)
  - New port to SWI-Prolog 6.0.2
- Changes:
  - License has been relaxed to GNU Lesser General Public License
  - New versions of command **`/debug_sql`** does not admit a traversing order yet (order option removed)

- Release notes of older DES versions are moved to the new document: **releasenotesDES.pdf**

- Fixed bugs:
  - Some spanned inputs without leading blanks in multi-line mode were not recognised
  - Duplicated object rules were retrieved several times
  - Some commands were not recognized in mixed or uppercase
  - Some listings in development mode did not display all rules
  - Some hypothetical queries led to exceptions
  - Existency of table and attributes in an **INSERT** SQL statement with a SQL data source was not checked
  - Parsing of a SQL relation separated by a leading space before the comma lead to syntax error
  - Predefined strong constraints relating a tuple of column names were rejected if its lexicographic order did not match the order in which they occur in table definition
  - Running info were logged
  - Some rules with conjunctions and disjunctions were not parsed correctly from consulted files
  - GNU Prolog source distribution stopped processing of batch files while encountering a **/shell** command
  - Predicate dependency graph and strata were not computed after issuing DML SQL statements **INSERT**, **DELETE** and DQL SQL statement **WITH**

1.29 Version 2.7 of DES (released on January, 3rd, 2012)

- Enhancements:
  - Extended relational algebra processor including all the original operators but division, and extended operators for dealing with outer joins, duplicate elimination, recursion, and grouping with aggregates
  - Multi-line input is also allowed in addition to the current single-line input. Long inputs as typical SQL statements can be spanned over several lines. When multi-line is enabled with the command **/multiline on**, Datalog inputs must end with a dot (.), and SQL and RA inputs with a semicolon (;). When disabled, each line is considered as a single (Datalog, SQL or RA) input and ending characters are optional
  - When multi-line input is enabled, remarks enclosed between **/*** and ***/** can span over several lines and can be nested as well
  - Single-line (**--**) and multi-line (**/******/**) remarks can be included in SQL statements at any place a separating blank can occur
  - SQL statement **CREATE TABLE** can include **LIKE** for creating a table with the same schema as an existing one

- SQL statement **DROP TABLE** can include **IF EXISTS** clause and can apply to a list of tables
- New (non-standard) SQL metadata statements (catalogued under ISL, Information Schema Language):
  - **SHOW TABLES;** List table names. *TAPI enabled*
  - **SHOW VIEWS;** List view names. *TAPI enabled*
  - **SHOW DATABASES;** List database names. *TAPI enabled*
  - **DESCRIBE *Relation*;** Display schema for ***Relation***, as **/dbschema** command does. *TAPI enabled*
- New commands:
  - **/list_tables** List table names. *TAPI enabled*
  - **/list_views** List view names. *TAPI enabled*
  - **/multiline** Display whether multi-line input is enabled
  - **/multiline *Switch*** Enable or disable multi-line input (**on** or **off** resp.)
  - **/ra** Switch to RA interpreter
  - **/ra *Query*** Execute an RA query
  - **/referenced_relations *Name*** Display relations directly referenced by a foreign key in ***Name***
  - **/referenced_relations *Name/Arity*** Display relations directly referenced by a foreign key in ***Name/Arity***
- Last line in a processed file must not end with a carriage return for its processing
- Faster abolish command and drop database SQL statement
- Display of the number of consulted constraints, if any
- Exceptions during constraint checking when consulting files are caught
- Faster parsing of Datalog rules and SQL statements
- A pivot variable that does not occur in the aggregate relation raises a syntax error
- Views are not required to be created with given column names
- Submitting a query or creating a view with duplicated columns is rejected
- Language command error messages instead of just "Input processing error"
- Improved compilation of **EXISTS** SQL clauses, using Datalog built-in **top**/2, which allows to prune the number of computed tuples

- Changes:
  - The system prompt for Datalog language changes to the old prompt **DES>**, as almost any input can be handled from this setting. The only inputs that

must explicitly submitted to a language processor are those that can be handled by several language processors

- Null identifiers are not wasted as eagerly as in previous versions
- Negation algorithm `et_not` do not longer rely on computations by strata
- New organization of system files:
  - `des_sql_debug.pl` (debugger extracted from former `des_sql.pl`)
  - `des_dl_debug.pl` (replaces `des_debug.pl`), and
  - `des_ra.pl` (includes RA processor)

- Fixed bugs:
  - Listings of SQL statements including Top-N queries failed
  - After submitting an incorrect SQL view, all of its temporary schema was not cleaned up
  - Variable names in consulted Datalog constraints were lost
  - New schema names defined in the list of local definitions inside a **`WITH`** or **`ASSUME`** SQL statement were not handled appropriately. Bug introduced in version 2.6
  - Only one blank was allowed after a **`SELECT`** statement. Bug introduced in version 2.6
  - Operator precedence in SQL conditions and Datalog bodies was not correctly handled (parentheses were needed to ensure correct operator applications)
  - Renamed relations could not be enclosed between parentheses
  - A renamed argument in a nested query was not visible for the **`WHERE`** condition of its outer query
  - Expressions in nested SQL queries could not be referenced from outermost queries
  - Type inference failed in some situations for equivalent internal string types (cf. `russell.sql`). Bug introduced in version 2.6
  - Underscored variables in a head rule made rule assertion fail
  - The Prolog interpreter did not handle conjunctive and disjunctive queries

1.30 Version 2.6 of DES (released on October, 26th, 2011)

- Enhancements:
  - A novel proposal for hypothetical SQL queries which allow to assume extra tuples in existing relations (either tables or views)
  - New SQL Top-N queries following ISO 2008 (another common form is also supported)

- New Datalog built-in **top**/2 for computing Top-N queries, i.e., those with the number of tuples in the answer limited to N
- SQL statements are allowed in the projection list, even as components of arithmetic expressions
- Anonymous variables are discarded from the answer schema should they occur in queries, views and autoviews (even in heads)
- New sub-graph algorithm for finding predicate dependency graphs restricted to queries. It replaces an older one with exponential complexity, which did consulting and/or querying some small programs to raise memory exhaust exceptions
- Display of predicate dependency graph is ordered
- Display of strata is first ordered by strata and then by predicate
- Running info display about number of inferred tuples, working with console and windows applications
- Datalog built-ins **distinct**/1 and **distinct**/2 also work for arbitrary queries, not only for atoms
- Enhanced solving performance by hash-indexing of extension table
- Enhanced time displays. Time is formatted as either milliseconds (MS) (MS ms.) for less than a second; or seconds (SS) and milliseconds (MS) (SS.MS s) for less than a minute; or minutes (MM), seconds and milliseconds (MM:SS.MS) for less than an hour; or hours, minutes, seconds and milliseconds (H:MM:SS) for greater than or equal to an hour
- Case-insensitive interactive user answers (debugging, test cases, ...)
- Keyboard interrupt caught for SWI-Prolog distributions. This allows interrupt the current computation without exiting DES
- Syntax error report on incompatible relation schemas in set operations
- Syntax error report about mismatch type for condition in metapredicate **group_by**
- Added hints on misspelled commands
- Help on commands and built-ins
- Hints on misspelled entries for:
  - Existing commands **/debug_datalog** and **/trace_datalog**
  - New commands **/pdg**
- Enabling TAPI for the next existing commands and synonyms:
  - **/consult**
  - **/reconsult**
  - **/cd**
  - **/pwd**
- New commands:

- **/check_db** Check database consistency w.r.t. declared integrity constraints (types, existency, primary key, candidate key, foreign key, functional dependency, and user-defined). Display a report with the outcome
- **/display_answer** Display whether display of computed tuples is enabled
- **/display_answer *Switch*** Enable or disable display of computed tuples (**on** or **off**, resp.) The number of tuples is still displayed
- **/hypothetical** Display whether hypothetical SQL queries are allowed
- **/hypothetical *Switch*** Enable or disable hypothetical SQL queries (**on** or **off**, resp.)
- **/indexing** Display whether hash indexing on extension table is enabled
- **/indexing *Switch*** Enable or disable hash indexing on extension table (**on** or **off**, resp.) Default is enabled, which shows a noticeable speed-up gain in some cases
- **/pdg *PredName*** Display the current predicate dependency graph restricted to the first predicate found with name ***PredName***
- **/pdg *PredName/Arity*** Display the current predicate dependency graph restricted to the predicate with name ***PredName*** and ***Arity***

- Changes:
  - Constraints assertions are not checked when disabling constraint checking
- Fixed bugs:
  - Integrity constraint checking could not be changed. Bug introduced in version 2.4
  - Syntax error exceptions when consulting files exited DES
  - A **FROM**-less SQL statement in a series of local view definitions of a **WITH** statement was not parsed
  - SQL parsing involving non-existent column names tweaks for discarding incorrect statements and accept correct statements
  - The command **/abolish** deleted rules of view definitions but not the schema
  - Datalog rule listings with added parentheses enclosing disjunctions when needed
  - Some nested SQL statements containing expressions were not parsed
  - SWI-Prolog distributions included incorrect computation time displays when detailed timing was enabled

- Some unsafe rules involving set variables were not transformed when safety transformation was enabled
- Use of set variables in equalities and `is`/2 always yielded to error messages, although the use were correct

1.31 Version 2.5 of DES (released on September, 13th, 2011)

- Enhancements:
  - A textual API for connecting DES with external systems. Several commands and queries can be read and answered using standard streams. Currently, TAPI-enabled queries and commands are those listed in Section 5.12, which are needed to interface database schemas and data to ACIDE [Sae07]. Queries include SQL DDL, DML, and DQL statements. Datalog constraint assertions and deletions are also supported
  - New Datalog (strong) integrity constraints: candidate key (uniqueness) and existency (forbid nulls) integrity constraints. Commands `/save_ddb` and `/restore_ddb` apply for such new constraints
  - Support for `UNIQUE` and `NOT NULL` column and table constraints in SQL `CREATE TABLE` statements
  - Added support to specify column names in SQL `INSERT INTO` statements
  - Nulls are no longer allowed in primary key columns
  - Type inferencing added to SQL DQL queries, in addition to the already supported DDL queries
  - Added type mismatch report error for ill-typed SQL statements
  - Answers from SQL queries are annotated with their inferred types
  - Limited-length types are also inferred for views and queries
  - Types returned by ODBC connections are labelled with their lengths
  - Tables and views are sorted in the result of command `/dbschema`
  - Column names are ordered in predefined integrity constraint displays
  - Enhanced SQL syntax error reporting for built-ins used as table and column identifiers
  - SQL syntax error reporting for unknown columns includes hints about similar column names, in addition to the already hints about table and view names
  - Commands involving table, view or relation names which are not defined provide hints
  - Hints on alternative names also include names with swapped characters
  - Trying to use a built-in symbol as a user identifier in a SQL statement is warned as a syntax error
  - Simplified error messages

- Extension table is not cleared when enabling duplicates. Instead, complete flag is reset, avoiding much recomputation
- New non-standard SQL statements **RENAME TABLE** and **RENAME VIEW**
- New commands:
  - **/tapi *Input*** Process ***Input*** and format its output for TAPI communication.
  - **/test_tapi** Test the current TAPI connection. *TAPI enabled*
  - **/drop_ic *Constraint*** Drop the specified integrity constraint, which starts with "**:-**"
  - **/dependent_relations *Relation*** Display the names of relations that depend on relation ***Relation***. *TAPI enabled*
  - **/dependent_relations *Relation/Arity*** Display in format *Name*/Arity those relations that depend on relation ***Relation/Arity***. *TAPI enabled*
  - **/is_empty *relation_name*** Display **$true** if the given relation is empty, and **$false** otherwise *TAPI enabled*
  - **/list_table_constraints *table_name*** List table constraints for ***table_name***. *TAPI enabled*
  - **/list_table_schemas** List table schemas. *TAPI enabled*
  - **/list_view_schemas** List view schemas. *TAPI enabled*
  - **/referenced_relations *Relation*** Display the name of relations that are directly referenced by a foreign key in relation ***Relation***. *TAPI enabled*
  - **/referenced_relations *Relation/Arity*** Display in format Name/Arity those relations that are directly referenced by a foreign key in relation ***Relation/Arity***. *TAPI enabled*
  - **/relation_exists *relation_name*** Display **$true** if the given relation exists, and **$false** otherwise. *TAPI enabled*
  - **/relation_schema *relation_name*** Display relation schema of ***relation_name***. *TAPI enabled*
  - **/sql_left_delimiter** Display the SQL left delimiter as defined by the current database manager (either DES or the external DBMS via ODBC) . *TAPI enabled*
  - **/sql_right_delimiter** Display the SQL right delimiter as defined by the current database manager (either DES or the external DBMS via ODBC) . *TAPI enabled*
- New port to SWI-Prolog 5.10.5
- New port to Ciao Prolog 1.14.2

- Changes:

- Identifier delimiters in output messages have been changed from `[` and `]` to `"`
- Either by consulting a file, or by dropping the database, or by abolishing the complete database imply to completely reset the database (Datalog rules, tables, views and constraint definitions are removed)

- Fixed bugs:
  - When enabling duplicates without clearing extension table, some duplicates were lost. Bug introduced in version 2.4
  - Datalog queries to ODBC connections failed when involving ground or aliased arguments
  - Some foreign key constraint were not properly checked against database before posting
  - Comma-separated arguments in commands were not always correctly parsed. In particular, this affected to consulting more than one file at a time
  - Command **`/dbschema`** did not always show all integrity constraints
  - Command **`/save_ddb`** incorrectly quoted ending dots in constraints
  - User-defined integrity constraints are syntactically identified (including variable names), therefore avoiding ambiguity for unifiable constraints
  - Several SQL delimited columns in the same statement were not correctly parsed
  - Creating an incorrect SQL view dealt to a table with the same name. Bug introduced in version 2.4
  - Most user-defined integrity constraints were not correctly parsed from files
  - When restoring the database, not all strong integrity constraints were removed
  - Write option in command **`log`** was not parsed

1.32 Version 2.4 of DES (released on July, 6th, 2011)

- Enhancements:
  - Safety and computability revisited for aggregate metapredicates. Most checks are moved to compile-time, covering also the new metapredicate **`distinct`**/2 and equality over evaluable expresssions
  - Added the Datalog tabled metapredicate **`distinct`**/2, which computes distinct outcomes for different values of given arguments and for a given relation. It takes effect when duplicates are enabled via the command **`/duplicates on`**
  - Comparison of expressions including null values are now supported. Two expressions are considered equivalent if they are *syntactically* equal. For instance, **`X=null,X+1=X+1`** succeeds, whereas **`X=null,Y=null,X+1=Y+1`** and **`X=null,X+1=1+X`** do not
  - Syntax error reporting about unbalanced parentheses in Datalog and SQL

- Syntax error reporting for metapredicate **group_by** involving incorrect use of variables in Datalog
- Simplified error reporting when syntax errors are detected
- Compilation of Datalog rules keep variable program names for exploded rules (way cool in development mode)
- Successive applications of **not**/1 are simplified instead of rewritten
- Negated calls to primitives are simplified by their complemented counterparts (e.g., **not(1<0)** is translated to **1>=0**). This in turn avoids the following null-related flaw: **not(null\=null)**, which should be semantically equivalent to **null=null**
- New commands:
  - **/running_info** Display whether running information (as the incremental number of consulted rules as they are read) is to be displayed
  - **/running_info** ***Switch*** Enable or disable display of running information (**on** or **off**, resp.)
  - **/rm** ***FileName*** Delete ***FileName*** from the file system
  - **/del** ***FileName*** Synonym for **/rm**
  - **/system** ***Goal*** Submit ***Goal*** to the underlying Prolog system (implementor's command)
- Internal null identifiers are reset whenever the database is cleared, and they otherwise start from 0 instead of 1
- Enabling (disabling) flags with commands **/compact_listings**, **/check**, **/development**, **/duplicates**, **/pretty_print**, **/safe**, **/simplification**, and **/verbose** warns should they are already enabled (disabled, resp.)
- New port to GNU-Prolog 1.4.0. Tested successfully for Ubuntu 10.04 and Windows 7
- New version of Windows GUI: ACIDE 0.8 with many improvements

- Changes:
  - Most errors regarding incorrect use of set variables are moved from run-time to compile-time
  - Unknown columns, tables and views are enclosed between double quotes
  - Datalog prompt is restored upon exception when processing a SQL statement
  - Internal representation of Datalog rules. Compiled rules are referenced by its rule identifiers in compilation roots, instead of storing full copies, therefore reclaiming less memory
  - Each rule has attached its textual variable names if they come from user inputs or instead they are automatically generated

- Showing Datalog compilations on the fly is also controlled by the command **/show_compilations**. Listings of compilations with the command **/listing** is still controlled by the command **/development**
- Showing running information is enabled by default. Such information display is not sent to the log, if enabled

- Fixed bugs:
  - Negated, compound calls involving either conjunction or disjunction were not correctly translated. Bug introduced in version 2.3
  - A Datalog 'having' condition with a variable to the right was incorrectly translated
  - Compound expressions including aggregate function **count**/0 were rejected
  - Parentheses in arithmetic expressions involving infix operators were not displayed when required
  - The listing command in development mode with pattern ***Name/Arity*** did not filter by ***Arity***
  - Evaluation of an expression containing a null returned a non ground null representation. This, for instance, made **X=null,Y=null,X+1=Y+1** true

1.33 Version 2.3 of DES (released on May, 24th, 2011)

- Enhancements:
  - SQL declarative debugger: Users can debug SQL views from a declarative debugging point-of-view. The system interactively asks questions to the user about relations involved in the computation of the debugged view. Trusted specifications add semantic references in order to cut the number of questions down
  - Added the Datalog tabled metapredicate **distinct**/1, which computes distinct outcomes for its argument, discarding duplicates. It takes effect when duplicates are enabled via the command **/duplicates on**
  - New Datalog functions and predicates for discarding duplicates along aggregation:
    - Aggregate functions: **count_distinct**/1, **count_distinct**/2, **avg_distinct**/2, **sum_distinct**/2, and **times_distinct**/2
    - Aggregate predicates: **count_distinct**/2, **count_distinct**/3, **avg_distinct**/3, **sum_distinct**/3, and **times_distinct**/3
  - Working **DISTINCT** and **ALL** keywords in SQL **SELECT** and **UNION** statements following SQL2 standard
  - Working **DISTINCT** and **ALL** keywords in SQL aggregate functions following SQL2 standard
  - Display of SQL statements compilations to Datalog, selectable with the new command **/show_compilations**

- Output from shell commands in windows applications are logged
- Compact listings can be enabled so that blank lines in the console output are removed
- New commands:
  - **/debug_sql** ***View*** **[*Options*]** Debug a SQL view, optionally specifying whether trusting tables or not, selecting a trust file and selecting a traverse order
  - **/show_compilations** Display whether compilations from SQL DQL statements to Datalog rules are to be displayed
  - **/show_compilations** ***Switch*** Enable or disable display of extended information about compilation of SQL DQL statements to Datalog clauses (**on** or **off**, resp.)
  - **/compact_listings** Display whether compact listings are enabled
  - **/compact_listings** ***Switch*** Enable or disable compact listings (**on** or **off**, resp.)
  - **/nospy** ***SPred*****[/*Arity*]** Removes the spy point on the given predicate in the host Prolog interpreter (implementor's command)
  - **/debug** Set debug mode in the host Prolog interpreter (implementor's command)
- Saving the deductive database to a file also includes constraints (type, existency, primary key, candidate key, functional dependency, foreign key, and user-defined). Restoring a database from file also recovers these constraints
- Added option **force** to command **/save_ddb**, which avoids asking the user should the file exists already
- Added help on **output** without argument
- Datalog predicate symbols and alphanumeric constants containing extended characters do not longer need to be enclosed between quotes
- SQL correlated queries are also allowed in comparison operators
- Escaped single quotes allowed in SQL strings
- SQL syntax error reporting for unknown tables, views and columns. Similar table and view names are provided for the user to choose
- Some run-time errors regarding incorrect use of built-ins (e.g., aggregates) are detected earlier during compilation
- Improved clpfd library for Ciao port, which allows to support type checking (thanks to Rémy Haemmerlé)
- New port to SWI-Prolog 5.10.4

- Changes:

- SQL natural joins keep the order of columns in the projection list, as usual in RDBMS implementations

- Fixed bugs:
  - Queries involving equality between nulls at the system prompt were incorrect. Bug introduced in version 2.1
  - When requesting help on a keyword which is both a command and a built-in (e.g., `log`), only help on command was displayed
  - Some synonyms in the help were not displayed
  - Type error raised when creating SQL views involving `TIMES` aggregate function
  - Single quotes inside quoted atoms were allowed
  - Predicate/Arity specifications in commands did not apply for arities greater than 9
  - Removing the compilation of nested applications of outer joins in SQL left some unremoved rules (when submitting SQL queries and dropping views)
  - Some safety warnings during compilation of some aggregate predicates were raised, even avoiding to solve some safe queries
  - After executing an `INSERT`, `DELETE` and `UPDATE` involving other (nested) SQL statements, the extension table was not cleared (so, an answer entry might occur in the table after finishing the modification statement)
  - Some incorrect SQL statements involving unexistent columns were not rejected

1.34 Version 2.2 of DES (released on March, 24th, 2011)

- Enhancements:
  - Type constraints can be imposed and checked on intensional predicates, not only on extensional ones
  - Improved type inference precision
  - Propositional relations can also be typed
  - Datalog type `char` added
  - Added alternative syntax for Datalog type constraints:

    `:- type(pred(col1:type1,...,coln:typen))` and
    `:- type(pred(type1,...,coln))`
  - Added Oracle predefined `'dual'` table
  - Added `FROM`-less SQL `SELECT` statements
  - Help system refactoring
  - New commands:
    - `/help Keyword` Display detailed help about command *Keyword*
    - `/apropos Keyword` Synonym for `/help Keyword`

        - `/prolog_system` Display the underlying Prolog engine version
    - Formatted ODBC error messages
    - From the Datalog input, `ALTER`, `USE` and `CREATE TABLE` SQL statements are also automatically sent to the opened ODBC datatabase, if a connection is already opened
    - Added warning for undeclared predicates occurring in basic queries (i.e., those predicates which have not been provided with either a defining rule or a type declaration)
    - When executing a query in development mode, its compilation is displayed
    - Multi-line remarks are allowed at the system prompt
    - Developer commands are now available in Ciao source distribution
    - New port to Ciao Prolog 1.13.0. Includes unreleased all-new `clpfd` library. This port replaces the old port to version 1.10p5
    - New port to SICStus Prolog 4.2.0, which includes enhanced and fixed ODBC library. Former limitations of DES w.r.t. this port are removed
    - In the context of an opened ODBC connection, predicate dependency graph and stratification are computed from the relational database schema, instead of querying each table and view. Computing the deductive database part does not change
    - More robust handling of ODBC exceptions
    - `SIGINT` interrupt signal is caught in SWI-Prolog version so that users can now interrupt DES (Ctrl-C usual keyboard interrupt)
    - SQL Server ODBC connections tested on spatial databases
- Changes:
    - Built-in Prolog DCG expansion replaced with an explicit translation, which can now be found in `des_dcg.pl`. This file is an adaptation of Ciao's `dcg_expansion.pl`. It works with all supported Prolog systems but GNU Prolog 1.3.1, which does not provide term expansion
    - A singleton anonymous variable denoted by an underscore in listings is displayed with an underscore. Up to version 2.1, it was given a name with letters, starting with `A`
    - Instantiation error exceptions coming from code implementing DES are now displayed (only useful for DES implementors)
    - Some program simplifications related to equalities have been omitted for the sake of type inferencing. This involves different source-to-source translations during query evaluation
    - Built-in `is`/2 is translated into `=`/2 at compilation-time when its right argument is already evaluated
    - Removed input error message from attempting to add types in the context of an opened ODBC connection
- Fixed bugs:

- Unsafety was not reported for anonymous variables (as, e.g., asserting the rule `p(_)`)
- Negation involving an aggregate or outer join predicate with atom syntactic form was not transformed (e.g., `not(count(p,0))`)
- Changing some system flags in no verbose mode displayed an info message
- Datalog types `char(N)` and `varchar(N)` were not parsed
- Some Datalog queries with duplicates enabled and an opened ODBC connection were incomplete in some special cases where rule identifiers collided
- Log recording upon exceptions repeated previous lines in some keyboard inputs
- Predicate dependency graph and stratification was not computed when closing an ODBC connection
- Some calls to predicates resulting from translating SQL queries involving disjunctions were incorrectly built, as in `select * from t where a=1 or b=1`

1.35 Version 2.1 of DES (released on November, 30th, 2010)

- Enhancements:
  - Access to Datalog relations from SQL statements. To this end, type declarations are provided to allow both give types and names to relation columns
  - Datalog (strong) integrity constraints: type, primary key, foreign key, functional dependencies and user-defined integrity constraints
  - SQL statements can be directly submitted from the Datalog prompt
  - Revised compilation of SQL views to Datalog rules, avoiding unnecesary intermediate relations
  - Enhanced performance: Built-in operators (`is`, `<`, `>`, ...) do not longer rely on the extension table mechanism, speeding computations up to ten times (cf. `fib(1000,X)` in `fib.dl`)
  - Negation can be applied to compound goals
  - Displaying of the number of consulted rules
  - Formatted Datalog syntax errors (error text, file name, line and column)
  - Updated manual
  - Output from the command `/builtins` rearranged in a way similar to `/help`
  - User inputs with trailing blanks after the ending optional dot are now accepted

- Now, string type constraints limit the length of strings as specified in their declarations (e.g., `char(1)` does not permit strings of length more than 1). Working but in Ciao Prolog source distribution
- Consulted Datalog files can contain multi-line remarks enclosed between `/*` and `*/`
- Reworked shell command. Outtput and error streams are redirected to the window application in MS Windows distros (this applies to GNU Prolog, SICStus Prolog and SWI-Prolog)
- New commands:
  - `/cat Filename` Type the contents of Filename. Also, the synonym `/type Filename` is provided
  - `/check` Display whether integrity constraint checking is enabled
  - `/check [Switch]` Enable or disable integrity constraint checking (`on` or `off`, resp.)
  - `/spy Pred[/Arity]` Set a spy point on the given predicate in the host Prolog interpreter (command intended for implementors, not users)
  - `/nospyall` Remove all Prolog spy points (command intended for implementors, not users)
  - `/t` Terminate the current DES session without halting the host Prolog system (command intended for implementors, not users)

- Changes:
  - For ODBC connections, the table `db_schema`, which is automatically created to have access to table and view names, was hiden in SICStus Prolog source version
  - Consulting an incorrect line in a Datalog program does not halt from reading subsequent files, if any
- Fixed bugs:
  - Empty strings were not displayed between single quotes
  - The empty constant (`''`) was rejected in Datalog rules
  - SICStus Prolog source version failed in retrieving tuples with Datalog queries in ODBC connections
  - Exiting without closing an ODBC connection raised an exception
  - Duplicated answers containing null values were not removed with duplicates disabled. Bug introduced in version 2.0
  - Character inputs were not displayed during batch processing as, e.g., answering single-character inputs ('`y`', '`n`', ...)
  - Ciao Prolog source distribution displayed incorrect paths when listing contents of directories containing '`..`'

- When duplicates were enabled, some recursive rules did not provide answers, as `fib(3)` in `fib.dl`
- Windows application entered a loop when closing the window with the white-crossed red button
- DES could not be interrupted via Ctrl-C in the Ciao source distribution
- Bitwise disjunction and conjunction were not correctly parsed

1.36 Version 2.0.1 of DES (released on September, 13th, 22nd, and October 7th, 2010)

- Enhancements:
  - DES 2.0.1 executable for Mac OS X Leopard (32bit, Intelx86).
  - DES 2.0.1 patches to SWI source distributions in order to support SWI-Prolog 5.10.1
- Fixed bugs:
  - DES 2.0.1 SICStus source version patched Datalog queries against ODBC connections

1.37 Version 2.0 of DES (released on August, 31st, 2010)

- Enhancements:
  - Connection to RDBs via ODBC connections (DSN providers as MySQL, MS Access, Oracle, ...) RDB tables and views can be queried both from SQL and Datalog
  - Duplicates are allowed in answers, both for Datalog and SQL
  - Datalog and SQL tracers
  - New commands:
    - `/open_db Name [Options]` Open and set the current ODBC connection to *Name*, where *Options*=[`user(Username)`] [`password(Password)`]. This connection must be already defined at the OS layer
    - `/close_db` Close the current ODBC connection
    - `/current_db` Display the current ODBC connection name and DSN provider
    - `/duplicates` Display whether duplicates are enabled
    - `/duplicates Switch` Enable or disable duplicates (`on` or `off`, resp.)
    - `/trace_sql View [Order]` Trace a SQL view in the given order (`postorder` or the default `preorder`)
    - `/trace_datalog Goal [Order]` Trace a Datalog basic goal in the given order (`postorder` or the default `preorder`)

        - **/output** ***Switch*** Enable or disable display output (**on** or **off**, resp.)
        - **/save_ddb** ***Filename*** Save the current Datalog database to a file
        - **/restore_ddb** ***Filename*** Restore the Datalog database in the given file (same as **consult**)
    - Results from **SELECT** SQL statements (those sent to an ODBC connection) can contain duplicates
    - Added **UPDATE** SQL statement
    - Added **varchar2** Oracle SQL datatype
    - Remarks can now start with '**--**', as in Oracle SQL
    - Both **EXCEPT** and **MINUS** are allowed to express SQL set difference
    - SQL user identifiers can be enclosed between quotation marks (either double quotes **""**, or square brackets **[]**, or backquotes **``**)
    - Closing the opened log file, if any, on quitting
    - Added timing information to SQL query processing, including listings which may include view processing from RDBs
    - Some dead code removal
- Changes:
    - New port to SICStus 4.*x*, which replaces the old port to SICStus 3.*x*
    - The command **debug** is renamed as **debug_datalog**
    - Executables have been built with SWI-Prolog, instead of SICStus Prolog
- Fixed bugs:
    - Asserting rules with a number as atom/string changed the type to number, as in **/assert t('1')**, which asserted **t(1)** instead of **t('1')**
    - Disjunctions in aggregate goals might lead to missing answers, as in **group_by((p(X,Y),(Y=a;Y=b)),[X],C=count)**
    - Some infix builtins were accepted without delimiting blanks, as **Xis1**, posed as a goal, and interpreted as **X is 1**
- Caveats and limitations:
    - See Section 10 of the user manual
- Known bugs:
    - The projection list of a natural outer join is not correct in all cases
    - Disjunctions in having conditions in the **group_by** clause may display errors which are not
    - Operator precedence in SQL conditions are not correctly handled. Use parentheses to ensure correct operator applications

1.38 Version 1.8.1 of DES (released on March, 17th, 2010)

- Fixed bugs:
  - The Windows and Linux executable distributions lacked some libraries regarding test case generation, which have been added in the current distributions
- Caveats and limitations:
  - See Section 10 of the user manual
- Known bugs:
  - The projection list of a natural outer join is not correct in all cases
  - Disjunctions in having conditions in the **group_by** clause may display errors which are not

1.39 Version 1.8.0 of DES (released on December, 18th, 2009)

- Enhancements:
  - An advanced test case generator supporting positive-negative, positive and negative test cases for views, ranging over integer and string data types
  - New command:
    - **/tc_size *Min Max*** Sets the minimum and maximum number of tuples generated for a test case
  - New use for existing command:
    - **/test_case *View [Options]*** Generates test case classes for the view ***View***. ***Options*** may include a class and/or an action parameters. The test case class is indicated by the values **all** (positive-negative), **positive**, or **negative** in the class parameter. The action is indicated by the values **display** (only display tuples), **replace** (replace contents of the involved tables by the computed test case), or **add** (add the computed test case to the contents of the involved tables) in the action parameter. Default parameters are **all** and **display**
  - More precise type inferring system
  - Enhanced syntax error reporting when consulting Datalog programs. An offending rule which is a valid term but is not a valid Datalog rule is listed together with location information
  - Enhanced pretty-print:
    - Rules: disjunctive bodies and quoted constants
    - SQL: indentation
    - **/dbschema**: bullets and expanded indentation
  - Informing that a goal cannot be debugged when its predicate is not defined
  - New switch for existing command:

        - `/timing detailed` Displays detailed elapsed time (parsing, computation, display and total elapsed times)
    - Line number information of consulted files is available also for the source distributions of both Ciao and SWI Prolog
- Changes:
    - The displayed integer type for tables and views has changed from `int` to `integer`
    - Any sequence of characters enclosed between quotes are allowed as a constant, as `'2*3'`
    - A bit more precise verbose output messages
- Fixed bugs:
    - Select statements with empty relations and `group_by` gave incorrect results
    - Translations of disjunctions in `group_by` conditions involving shared variables were incorrect
    - Some output displays were not logged via the command `/log`
    - Rule retraction may behave incorrectly when compiled rules cannot be differentiated
    - When a set of tables were dropped, their foreign keys were not
    - A renaming in the projection list of a SQL statement with the same identifier as input relations was incorrectly translated
    - Dropping and recreating a view failed to delete the defining Datalog rules for the rule, raising a warning
    - Removed meaningless warning message when redefining a table
    - Consulting a datalog program with syntax errors when safety is enabled yielded a loop
    - When asserting a rule and simplification enabled, the correct variable names where not displayed in the translation in some cases
- Caveats and limitations:
    - See Section 10 of the user manual
- Known bugs:
    - The projection list of a natural outer join is not correct in all cases
    - Disjunctions in having conditions in the `group_by` clause may display errors which are not

1.40 Version 1.7.0 of DES (released on October, 30th, 2009)

- Enhancements:
    - Extended SQL grammar and processor to cope with types as well as table and column constraints (primary key and foreign key)

- Type system for SQL. Primitive types include: `char`, `char(n)`, `varchar(n)`, `varchar`, `string`, `int`, `integer`, and `real`
- Basic type checking/inferring system for SQL views. Inferred types for views are displayed via `/dbschema` and, for autoviews, in the answer relation. Inferring precision is low (the types of expressions and numbers are not inferred)
- Domain, primary key, and referential integrity constraints for tables created with SQL statements
- Datalog aggregate predicates: `group_by/3`, `count/3`, `count/2`, `sum/3`, `times/3`, `avg/3`, `min/3`, and `max/3`
- Datalog aggregate functions: `count/0`, `count/1`, `sum/1`, `times/1`, `avg/1`, `min/1`, and `max/1`
- Datalog predicate builtins: `is_null/1` and `is_not_null/1` for determining whether their single argument is a null value or not, respectively
- Test case generation for views
- New commands:
  - `/test_case View` Generates all test case classes of for the given view
  - `/p Filename` Shorthand for `/process Filename`
- Upgraded commands:
  - `/listing Head` Lists all rules whose heads are subsumed by `Head`
  - `/listing Head:-Body` Lists all rules that are subsumed by `Head:-Body`
- The command `process` looks for its input filename, allowing to omit the extensions `.sql` and `.ini`
- Comparison operators can include arithmetic expressions, as in `A<2*B`. This also means that equality behaves more generally than `is/2`, as shown in the query `sqrt(2)=X`, which returns `{ sqrt(2) = 1.4142135623730951 }`
- Some arithmetic expressions are precomputed when translating SQL statements to Datalog rules
- Displaying the number of tuples in rule listings, retracts, and abolishes
- Adding development flag status to the listing of `/status`

- Changes:
  - A table definition with a `CREATE TABLE` statement must include a type for each attribute. Former table definitions (up to version 1.6.2) are no longer valid
  - Evaluation of an arithmetic expression including a null value returns a null, instead of raising an exception

  - Operands of comparison operators are evaluated. Only arithmetic expressions are allowed, up to now. So, `X=Y+2` is allowed whenever Y is bound
  - The distribution files `des1.pl`, `dessql.pl`, and `desdebug.pl` have been renamed to `des_glue.pl`, `des_sql.pl`, and `des_debug.pl`, respectively
- Fixed bugs:
  - Development listings via `/dbschema` were not displaying compiled Datalog rules
  - String constants including only digits were incorrectly parsed as numbers
  - Failed to parse SQL set statements involving constants in the projection list
  - Nulls were not correctly read from files
  - `IS NULL` and `IS NOT NULL` in SQL statements were not behaving correctly
  - Safety checks involving disjunctions were not always properly performed, as in `p(X) :- q(X);r(X)`
  - The command `/operators` was never implemented but listed via `/help`. It has been removed
  - Listings of exploded rules were not displaying the correct source variable names in bodies
  - Some rules could not been asserted under simplification, as `p(X):-X=1;X=2`
  - Error when a multiply renamed table occurs in a SQL statement, as in `select * from t t1,t t2 where t1.a=t2.a`
- Caveat:
  - Batch processing cannot be nested
- Known bugs:
  - The projection list of a natural outer join is not correct in all cases
  - Disjunctions in having conditions in the `group_by` clause may yield to errors which are not

1.41 Version 1.6.2 (released on March, 10th, 2009)

- Enhancements:
  - Null values has been included both for Datalog programs and SQL statements
  - Novel outer join Datalog functions: `lj/3`, `rj/3`, and `fj/3`
  - Outer join SQL clauses added: `LEFT [OUTER] JOIN`, `RIGHT [OUTER] JOIN`, and `FULL [OUTER] JOIN`

- Solving algorithm enhanced for stratified queries. Partial recomputations of lower-stratum predicates are avoided
- Compilation of SQL **WHERE** conditions to Datalog rules now provides shorter and more efficient programs
- Disjunctions in Datalog rule bodies
- New commands:
  - **/development** *Switch* Enables/Disables development listings. These listings show the source-to-source translations needed to handle null values, Datalog outer join built-ins, and disjunctive literals
  - **/development** Displays whether development listings are enabled
  - **/simplification** *Switch* Enables/Disables simplification of Datalog rules. Rules with equalities, **true**, and **not(***BooleanValue***)** are simplified
  - **/simplification** Displays whether rule simplification is enabled
- **WHERE** conditions accept arithmetic expressions (e.g., **1+t.a>3**)
- Display of the number of undefined computed tuples, and the number of tuples in the extension table answer and call sets
- Parentheses in Datalog rule bodies, not only in arithmetic expressions, are allowed
- Parentheresed listings of Datalog rule bodies, making more readable bodies with conjunctions and disjunctions
- Simplification of rules containing equalities

- Changes:
  - Rule listings are grouped by predicate name and arity. For a given predicate name and arity, facts come first, followed by rules with right hand sides. The order of facts and rules follows Prolog standard order between terms
  - Datalog rules resulting from translating views change the naming convention to (the more readable) *ViewName_Arity_Number* in lieu of *ViewName***$p***Number*
  - Results from Datalog autoviews are given the relation name **answer** instead of **autoview**
  - Pretty-print is applied to all Datalog rule listings
  - Safety warnings are not hidden by computability warnings
- Fixed bugs:
  - Unformatted SQL statement display for certain conditions and joins
  - Parsing error for **EXISTS** clause (no blanks between **EXISTS** and opening parenthesis were allowed)
  - SQL arithmetic functions could only be written in lowercase

- Some **WHERE** conditions incorrectly translated into Datalog conditions (bug introduced in version 1.6.1)
- Some **WHERE** conditions involving parentheses incorrectly parsed
- Correlated SQL queries with non-basic conditions were incorrectly translated into Datalog rules
- **DELETE** SQL statements failed to be parsed (copy-paste bug introduced in version 1.6.1)
- Some unsafe Datalog queries were not rejected for computation (as **X=Y**)
- During startup batch processing of **des.ini**, some tasks upon exceptions were not performed
- Typing **des.** in a Prolog interpreter after abnormally quitting the system did not result in exception catching anymore
- A class of unsafe rules was not be able to be preprocessed for reordering body goals, yielding non-termination
- Incomplete error message

- Known bugs:
  - The projection list of a natural outer join is not correct in all cases
- Caveat:
  - Computable SQL statements follow the grammar in the manual. The current grammar parses extra clauses which cannot be computed yet (e.g., **ORDER BY**, ...)

1.42 Version 1.6.1 (released on November, 10th, 2008)

- Enhancements:
  - Arithmetic expressions are allowed in the projection list of **SELECT** statements
  - Subqueries in comparisons (**=**, **<**, **>**, ...), in either side or even in both sides of the comparison operator (read as **ANY**, not **ALL**, which is unsupported up to now)
  - Display of the number of computed, inserted and deleted tuples
  - Commands are case-insensitive
  - Some tweaks on the SQL parsing code for making it hopefully more understandable and efficient
  - The answer to a SQL query is a relation with name **'answer'**, and its schema is displayed when solving it
  - A new use for the **/dbschema** command: Now, it accepts an optional argument (a database object, which can be a view or a table name) for restricting the displayed schema
  - The **/dbschema** command informs about local view definitions for each view

- A new SQL DDL statement: `drop database`, which drops the database (including tables, views, and rules)
- Stratifications are not computed during building a view that involves local views. As a consequence, several messages are suppressed (as 'undefined' and 'non stratifiable')

- Changes:
  - Inserted and deleted tuples are not shown
- Fixed bugs:
  - Complex left-hand-side relations in joins failed to be parsed
  - Conjunctive Prolog goals failed to be parsed (bug introduced in version 1.6.0)
  - Natural joins now return common attributes only once
  - Datalog rules involving expressions with (prefix) unary operators were incorrectly displayed as infix
  - Parsing of Datalog bodies failed is some situations were arithmetic operators were involved (as in `/assert p(X) :- X is -(1)`)
  - Parsing of projection lists failed in some situations where `table.*` was intermixed with references to single table attributes
  - Program transformation for obtaining safe rules yielded incorrect results in some cases
  - When dropping a view, its local view definitions (if any) were not dropped as well
  - Different views could define the same local view name
  - `/listing Name` failed to list rules of different arities (bug introduced in version 1.6.0)

1.43 Version 1.6.0 (released on July, 28th, 2008)

- Enhancements:
  - SQL query language added to the system: DDL (Data Definition Language), DML (Data Manipulation Language), and DQL (Data Query Language)
  - Common database for different query languages. Relations defined via SQL or via Datalog can be interchangeably accessed by queries in any language
  - Pretty-print listings for Datalog programs and SQL statements
  - Processing of batch files via the new command `/process File`
  - Display of 'File not found' errors
  - Lexicographically ordered listings
  - New commands:
    - `/datalog` Switches to Datalog interpreter

- **`/datalog`** ***`Query`*** Executes a Datalog query
- **`/prolog`** Switches to Prolog interpreter
- **`/sql`** Switches to SQL interpreter
- **`/sql`** ***`Query`*** Executes a SQL query
- **`/dbschema`** Displays the database schema
- **`/pretty_print`** Displays whether pretty print for listings is enabled
- **`/pretty_print`** ***`Switch`*** Enables/Disables pretty print
- **`/process`** ***`File`*** Processes the contents of ***`File`*** as if they were typed at the system prompt

- Changes:
  - Changed some output formatting for the debugger
  - Some tweaks on system messages, mainly referring to safety/computability
  - Initial status: Program transformation and time display are disabled by default
  - System status is listed at start-up
  - Listings of Datalog rules are ordered
- Fixed bugs:
  - The debugger in SICStus-based releases yielded incorrect results
  - Asserting/Consulting some unsafe clauses without program transformation yielded failure, raising an input error /failing to consult

1.44 Version 1.5.0 (released on December, 30th, 2007)

- Enhancements:
  - A more fine-grained debugging as long as individual clauses can be inspected
  - Warning and error messages provided for:
    - Undefined predicates which are called by rules each time the database is changed
    - Unsafe rules
    - Execution exceptions known at compile-time
  - Exception messages provided for:
    - Execution exceptions unknown at compile-time
  - Rule transformation for allowing computation of safe rules which may raise run-time exceptions due to built-ins
  - Rejection of unsafe or uncomputable queries, views and autoviews

- Catching of instantiation errors
- Rule source annotated for debugging and informative errors, i.e., file and lines in the program (if consulted) or assertion time (if manually asserted)
- Elapsed time display
- New basic, simpler (although less efficient than the already implemented) algorithm for computing stratified negation, following [SD91]
- Fresh variables are given new variable names instead of numbers
- New commands:
  - **`/negation`** Displays the selected algorithm for solving negation
  - **`/negation`** ***`Algorithm`*** Sets the required Algorithm for solving negation (**`strata`** or **`et_not`**)
  - **`/timing`** Displays whether elapsed time display is enabled
  - **`/timing`** ***`Switch`*** Enables or disables elapsed time display (**`on`** or **`off`**, resp.)
  - **`/safe`** Displays whether program transformation is enabled
  - **`/safe`** ***`Switch`*** Enables or disables program transformation (**`on`** or **`off`**, resp.)
- Changed commands:
  - **`/verbose`** Displays whether verbose output is enabled
  - **`/verbose`** ***`Switch`*** Enables or disables verbose output messages (**`on`** or **`off`**, resp.)
- Deprecated commands:
  - **`/noverbose`**
- Slight modifications on existing commands:
  - **`/debug`** ***`Goal Level`*** The inspection level can be set with the second optional argument with **`p`** for predicate level and **`c`** for clause level
  - **`/status`** Now, it also displays the selected algorithm for negation and whether program transformation is enabled
  - **`/version`** For matching the 'standard' display
- New examples added to the directory **`examples`**
- The Prolog database corresponding to the Datalog loaded programs has been discarded, therefore using only one representation for them
- Revised and upgraded user's manual

- Changes:
  - Inequality built-ins cause an error and stops execution whenever they are computed with any non-ground argument (formerly, they silently failed)
- Fixed bugs:

- The Linux version did not work. Now, it has been fixed and tested on Ubuntu 6.10, Kubuntu 7.04 (Feisty), and Mandriva Linux 2007 Spring
- The parser did not detect that the argument of `not` could be a variable
- Name clashes when loading programs and asserting rules are avoided

1.45 Version 1.4.0 (released on September, 2nd, 2007)

- Enhancements:
  - Arithmetic has been added. The infix builtin 'is' allows the evaluation of arithmetic expressions
  - Arithmetic operators:
    - `\` Bitwise negation
    - `-` Negative value of its single argument
    - `**` Power
    - `^` Synonym for power
    - `*` Multiplication
    - `/` Real division
    - `+` Addition
    - `-` Subtraction
    - `//` Integer quotient
    - `rem` Integer remainder
    - `\/` Bitwise disjunction between integers
    - `#` Bitwise exclusive or between integers
    - `/\` Bitwise conjunction between integers
    - `<<` Shift left the first argument the number of places indicated by the second one
    - `>>` Shift right the first argument the number of places indicated by the second one
  - Arithmetic functions:
    - `sqrt` Square root
    - `log` Natural logarithm of its single argument
    - `ln` Synonym for `log/1`
    - `log` Logarithm of the second argument in the base of the first one
    - `sin` Sine
    - `cos` Cosine
    - `tan` Tangent

- **cot** Cotangent
- **asin** Arc sine
- **ac**
- **os** Arc cosine
- **atan** Arc tangent
- **acot** Arc cotangent
- **abs** Absolute value
- **float** Float value of its argument
- **integer** Closest integer between 0 and its argument
- **sign** Returns -1 if its argument is negative, 0 otherwise
- **gcd** Greatest common divisor
- **min** Least of two numbers
- **max** Greatest of two numbers
- **truncate** Integer part as a float
- **float_integer_part(*X*)** Integer part as a float
- **float_fractional_part(*X*)** Fractional part as a float
- **round** Closest integer
- **floor** Greatest integer less or equal to its argument
- **ceiling** Least integer greater or equal to its argument

- Arithmetic constants:
  - **pi** Archimedes' constant
  - **e** Euler's number
- Scientific notation supported
- Autoviews (automatic temporary views) for conjunctive queries on the fly
- Parsing of programs, queries, and asserted rules
- New command:
  - **/status** Displays the current status of the system
- Output from the command **/builtins** has been rearranged
- Upgraded input error message
- Prolog goals (submitted via the command **/prolog**) can be conjunctive goals
- Revised and upgraded user's manual
- Revised and homogeneized input processing
- Line comments (starting with **%**) are allowed as prompt inputs (useful for commenting lines in batch files)

  - File and path names enclosed between single quotes for error reporting in OS commands (therefore clarifying misusing of blanks)

- Fixed bugs:
  - Underscores in variables were incorrectly parsed
  - Asserted rules had missing program variable names
  - The output stream was not flushed when prompting user input in the debugger and when prompting new Prolog solutions using `/prolog`
  - File and directory names as numbers threw an exception in OS commands
  - Incorrect goal when abolishing no rules
  - Some commands did not admit blanks between arguments
  - Fixed some disarranged displays
  - Batch processing tried to open both `.ini` and `.pl` files
  - Dangling choice points in several places
  - Anonymous variables were incorrectly parsed
  - Debugging was not possible during batch processing

1.46 Version 1.3.0 (released on May, 2nd, 2007)

- Enhancements:
  - Declarative debugger
- Fixed bugs:
  - The output stream was not flushed before waiting the user input. This presented a connection problem with the configurable IDE ACIDE (See Contributions)
- Contributions:
  - ACIDE (A Configurable Development Environment). Authors: Diego Cardiel Freire, Juan José Ortiz Sánchez, and Delfín Rupérez Cañas, leaded by Fernando Sáenz. 3/2007. Description: This project is aimed to provide a multiplatform configurable integrated development environment which can be configured in order to be used with any development system such as interpreters, compilers and database systems. Features of this system include: project management, multifile editing, syntax colouring, and parsing on-the-fly (which informs of syntax errors when editing programs prior to the compilation). Status: alpha.
  - Emacs development environment. Author: Markus Triska. 22/2/2007. Description: Provides an integration of DES into Emacs. Once a Datalog file has been opened, you can consult it by pressing F1 and submit queries and commands from Emacs.

1.47 Version 1.2.0 (released on February, 9th, 2007)

- Enhancements:

- Solving-by-stratum algorithm
- Temporary views, which allow to write a temporary rule whose head is solved as a query
- Program variable names are kept to allow more readable program listings
- Syntax error reports when loading programs in standalone applications and SICStus source distribution
- Handling and reporting of Prolog exceptions in standalone applications, SWI and SICStus source distribution
- New commands:
  - **`/verbose`** (default option) for verbose output
  - **`/noverbose`** for abbreviated messages
  - **`/strata`** displays the current stratification
  - **`/pdg`** displays the current predicate dependency graph
  - **`/dir`** synonym of **`/ls`**
  - **`/log`** ***`FileName`*** sets the current log to ***`FileName`***
  - **`/log`** displays the current log file, if any
  - **`/nolog`** disables logging
  - **`/version`** for displaying the current system version
- New uses for existing command: **`/abolish`** ***`Name`***, and **`/abolish`** ***`Name/Arity`***
- Batch processing
- Rearranged and revised help
- Reworked command and query-related messages
- Consulting/Reconsulting files avoids duplicates
- Added examples

- Fixed bugs:
  - Loading an incorrect Datalog program exited standalone applications (**`des.exe`** and **`deswin.exe`** applications)
  - Evaluating Prolog goals via **`/prolog`** failed for programs containing negation
  - For several commands, blanks between a command and its arguments were not consumed but the first one
  - Non existent directory errors were not caught in command **`/ls`**

1.48 Version 1.1.2 (released on December, 20th, 2006)

- Enhancements:
  - New uses for existing commands: **`/list_et`** ***`Name`***, **`/listing`** ***`Name`***

- Fixed bugs:
  - The commands **`/list_et`** and **`/clear_et`** were not properly parsed
  - Infix operators allowed a variable argument

1.49 Version 1.1.1 (released on February, 21st, 2005)

- A new executable version for Linux
- Enhancements:
  - Atoms can contain blanks
- Fixed bugs:
  - When using **`/prolog`**, DES1.1 did not find predicates defined without facts.

1.50 Version 1.1 (released on March, 4th, 2004)

- Full recursion
- Memoization techniques
- Gathering of undefined facts under non stratified programs (incomplete algorithm)
- Several new commands:
  - **`/listing`** ***`Name/Arity`***. Lists Datalog rules matching the pattern
  - **`/retractall`** ***`Head`***. Deletes all Datalog rules matching head
  - **`/list_et`**. Lists contents of the extension table
  - **`/list_et`** ***`Name/Arity`***. Lists contents of the extension table matching the pattern
  - **`/clear_et`**. Clears the extension table
  - **`/builtins`**. Lists built-in operators
  - **`/cd`** ***`Path`***. Sets the current directory
  - **`/cd`**. Sets the current directory to the directory where DES was started from
  - **`/pwd`**. Displays the current directory
  - **`/ls`**. Displays the contents of the current directory
  - **`/ls`** ***`Path`***. Displays the contents of the given absolute or relative directory
  - **`[`*`FileNames`*`]`**. Consults a list of Datalog files abolishing previous rules
  - **`[+`*`FileNames`*`]`**. Consults a list of Datalog files keeping previous rules
  - **`/shell`** ***`Command`***. Submits a command to the operating system shell
- Cosmetic changes:
  - Commands start with a slash
  - Command arguments are no longer enclosed in brackets
  - Both commands and queries may end with a dot

- Fixed bugs:
    - Primitives fail adequately when they should do it, instead of exiting from the interpreter

1.51 Version 1.0 (released on December, 2003)

Version 1.0 of DES, the first public release of the system, featured:

- Naïve Datalog system intended to be complete w.r.t. relational algebra
- Limited support for recursion: Termination is not guaranteed for some recursive programs
- Basic Negation
- Built-in Operators
    - `=` Syntactic equality
    - `\=` Syntactic disequality
    - `>` Greater than
    - `>=` Greater than or equal to
    - `<` Less than
    - `=<` Less than or equal to
    - `not(Goal)` Negation
- Commands
    - `consult(File)` Loads a Datalog program abolishing current rules
    - `reconsult(File)` Loads a Datalog program keeping current rules
    - `assert((Head:-Body))` Asserts a new rule
    - `retract((Head:-Body))` Retracs a rule
    - `abolish` Abolishes the loaded program
    - `listing` Lists the loaded rules.
    - `prolog(Goal)` SLD execution of Goal.
    - `halt` Quits the system
    - `help` Displays the help

2. Documentation License

GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be

listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their

titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with … Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being
LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.